Practical 1

Aim:(i) Create tables according to the following definition

TABLE 1: Salespeople

Code :

create table Salespeople (Snum number(10), sname varchar(50), city varchar(50), comm decimal(20,20))

desc Salespeople

insert into Salespeople values(1001,'Peel','London',0.12)

insert into Salespeople values(1002,'Serres','San jose',0.13)

insert into Salespeople values(1003,'Axelord','New York',0.10)

insert into Salespeople values(1004,'Motika','London',0.11)

insert into Salespeople values(1005,'Rifkin','Barcelona',0.15)

select *from Salespeople

Output:

SNUM	SNAME	CITY	COMM
1001	Peel	London	.12
1002	Serres	San jose	.13
1003	Axelord	New York	.1
1004	Motika	London	.11
1005	Rifkin	Barcelona	.15

5 rows returned in 0.00 seconds CSV Export

TABLE 2: Customer

Code :

create table Customer(Cnum number(10), Cname varchar(50), City varchar(50), Raking number(10), Snum number(10))

desc Customer1

insert into Customer values(2001,'Hoffman','London',100,1001)

insert into Customer values(2002,'Giovamne','Rome',200,1003)

insert into Customer values(2003,'Liu','San jose',300,1002)

insert into Customer values(2004,'Grass','Berlin',100,1002)

insert into Customer values(2006,'Clemens','London',300,1007)

insert into Customer(Cnum, Cname, Raking, Snum) values(2007,'Pereria',100,1004)

select *from Customer

Output:

CNUM	CNAME	CITY	RAKING	SNUM
2001	Hoffman	London	100	1001
2002	Giovamne	Rome	200	1003
2003	Liu	San jose	300	1002
2004	Grass	Berlin	100	1002
2006	Clemens	London	300	1007
2007	Pereria	-	100	1004

6 rows returned in 0.00 seconds CSV Export

TABLE 3: Orders

Code :

create table Order(Onum number(10), Amount number(4,2), Odate date, Cnum number(10), Snum number(10))

desc Order

insert into Order values(3001,18.96,'10-MAR-2008',2002,1002)

insert into Order values(null,null,null,null,null)

insert into Order values(null,null,null,null,null)

select *from Order

Output :

ONUM	AMOUNT	ODATE	CNUM	SNUM
3001	18.96	10-MAR-08	2002	1002
-	-	-	-	-
-	-	-	-	-

3 rows returned in 0.00 seconds CSV Export

- **Based on the given table perform following queries:**
  1. **Display snum, sname, city and comm. Of all salespeople.**

     Query : select *from Salespeople

     Output:

SNUM	SNAME	CITY	COMM
1001	Peel	London	.12
1002	Serres	San jose	.13
1003	Axelord	New York	.1
1004	Motika	London	.11
1005	Rifkin	Barcelona	.15

5 rows returned in 0.00 seconds CSV Export

2. **Display all snum without duplicates from all orders.**

   Query: select distinct Snum from Order4

   Output:

SNUM
-
1002

2 rows returned in 0.00 seconds

3. **Display names and commissions of all salespeople from London.**

   Query: select sname, comm from Salespeople where city = 'London'

   Output:

SNAME	COMM
Peel	.12
Motika	.11

2 rows returned in 0.00 seconds

4. **All customers with a rating of 100.**

   Query : select *from Customer1 where Raking = 100

   Output:

CNUM	CNAME	CITY	RAKING	SNUM
2001	Hoffman	London	100	1001
2004	Grass	Berlin	100	1002
2007	Pereria	-	100	1004

3 rows returned in 0.00 seconds CSV Export

5. **Produce order no, amount and date for all rows in the order table.**

   Query: select Onum, Amount, Odate from Order4

   Output:

ONUM	AMOUNT	ODATE
3001	18.96	10-MAR-08
-	-	-
-	-	-

3 rows returned in 0.00 seconds

6. **All customers who were either located in San Jose or had a rating above $200.**

   Query: select *from Customer1 where Raking>200 or city = 'San jose'

   Output :

CNUM	CNAME	CITY	RAKING	SNUM
2003	Liu	San jose	300	1002
2006	Clemens	London	300	1007

2 rows returned in 0.00 seconds CSV Export

7. **All customers in San Jose, who have a rating > 200.**

Query : select *from Customer1 where Raking>200 and city = 'San jose'

Output:

CNUM	CNAME	CITY	RAKING	SNUM
2003	Liu	San jose	300	1002

1 rows returned in 0.00 seconds CSV Export

8. **All orders for more than $1000.**

Query: select *from Order4 where Amount>1000

Output: Results Explain Describe Saved SQL History

no data found

9. **Names and cities of all salespeople in London with a commission above 0.10.**

Query: select sname, city from Salespeople where comm>0.10

Output:

SNAME	CITY
Peel	London
Serres	San jose
Motika	London
Rifkin	Barcelona

4 rows returned in 0.01 seconds CSV Export

10. **All customers excluding those with rating <= 100 unless they are located in Rome.**

Query: select *from Customer1 where Raking>100 or city = 'Rome'

Output:

CNUM	CNAME	CITY	RAKING	SNUM
2002	Giovamne	Rome	200	1003
2003	Liu	San jose	300	1002
2006	Clemens	London	300	1007

3 rows returned in 0.00 seconds CSV Export

11. **All salespeople either in Barcelona or in London.**

Query: select *from Salespeople where city in('Barcelona' ,'London')

Output:

SNUM	SNAME	CITY	COMM
1001	Peel	London	.12
1004	Motika	London	.11
1005	Rifkin	Barcelona	.15

3 rows returned in 0.01 seconds CSV Export

12. All customers without a city.

Query: select *from Customer1 where city is null

Output:

CNUM	CNAME	CITY	RAKING	SNUM
2007	Pereria	-	100	1004

1 rows returned in 0.00 seconds CSV Export

13. All orders taken on oct. 3rd or 4th 1994.

Query: select *from Order4 where Odate between '03-OCT-1994' and '04-OCT-1994'

Output:

Results Explain Describ

no data found

ii)Write the following simple SQL Queries on the University Dataset:

1. **Find the names of all the students whose total credits are greater than 100**

Query: select name from student where total credits > 100

2.Find the course id and grades of all courses taken by any student named 'Tanaka'

Query: select id, grades from courses where student = 'Tanaka'

3.Find the ID and name of instructors who have taught a course in the Comp. Sci. department, even if they are themselves not from the Comp. Sci. department. To test this query, make sure you add appropriate data and include the corresponding insert statements along with your query.

Query: select id , name from course where department = 'Computer Science Department'

4.Find the courses which are offered in both 'Fall' and 'Spring' semester (not necessarily in the same year).

Query: select courses from record where semester = 'Fall' and semester = 'Spring'

CONCLUSION: In this practical I learned about creating tables using SQL commands and select query.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 2

Aim:(i) Create the below given table and insert the data accordingly.

Create Table Job (job_id, job_title, min_sal, max_sal)

Code :

```
create table job(job_id varchar2(15),job_title varchar2(30), min_sal number(7,2),max_sal number(7,2))
insert into job values('it_prog','Programmer',4000,10000)
insert into job values('mk_mgr','Marketing manager',9000,15000)
insert into job values('fi_mgr','Finance manager',8200,12000)
insert into job values('fi_acc','Account',4200,9000)
insert into job values('lec','Lecturer',6000,17000)
insert into job values('comp_op','Computer Operator',1500,3000)
select * from job
```

Output:

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL
it_prog	Programmer	4000	10000
mk_mgr	Marketing manager	9000	15000
fi_mgr	Finance manager	8200	12000
fi_acc	Account	4200	9000
lec	Lecturer	6000	17000
comp_op	Computer Operator	1500	3000

Create table Employee (emp_no, emp_name, emp_sal, emp_comm, dept_no, l_name, dept_name, job_id, location, manager_id, hiredate) and insert following values in the table Employee.

Code:

```
create table employee(emp_no number(3),emp_name varchar2(30),emp_sal number(8,2),emp_comm number(6,1),dept_no number(3), l_name varchar2(30), dept_name varchar2(30), job_id varchar2(15), location varchar2(15), manager_id number(5), hiredate date)

insert into employee values(101,'Smith',800,'',20,'shah','machine learing','fi_mgr','toronto',105,'09-aug-96')
insert into employee values(102,'Snehal',1600,300,25,'gupta','data science','lec','las vegas','','14-mar-96')
insert into employee values(103,'Adama',1100,0,20,'wales', 'machine learing', 'mk_mgr','ontario', 105, '30-nov-95')
insert into employee values(104,'Aman',3000,'',15,'sharma','virtual reality','comp_op','mexico',12,'02-oct-97')
```

insert into employee values(105,'Anita',5000,50000,10,'patel','big data analysis','comp_op','germany', 107,'01-jan-98')

insert into employee values(106,'Sneha',2450,24500,10,'joseph','big data analytics','fi_acc', 'melbourne', 105,'26-sep-97')

insert into employee values(107,'Anamika',2975,'',30,'jha','artificial intelligence','it_prog','new york', '','15-jul-97')

select * from employee

Output:

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
101	Smith	800	-	20	shah	machine learning	fi_mgr	toronto	105	09-AUG-96
102	Snehal	1600	300	25	gupta	data science	lec	las vegas	-	14-MAR-96
103	Adama	1100	0	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95
104	Aman	3000	15	12	sharma	virtual reality	comp_op	mexico	2	02-OCT-97
105	Anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98
106	Sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97
107	Anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97

Create table deposit(a_no,cname,bname,amount,a_date) and Insert following values in the table deposit.

Code:

create table deposit(a_no varchar2(5), cname varchar2(15),bname varchar2(10),amount number(7,2),a_date date)

insert into deposit values(101,'Anil','andheri',7000,'01-jan-06')

insert into deposit values(102,'Sunil','virar',5000,'15-jul-06')

insert into deposit values(103,'Jay','villeparle',6500,'12-mar-06')

insert into deposit values(104,'Vijay','andheri',8000,'17-sep-06')

insert into deposit values(105,'Keyur','dadar',7500,'19-nov-06')

insert into deposit values(106,'Mayur','borivali',5500,'21-dec-06')

select *from deposit

Output:

A_NO	CNAME	BNAME	AMOUNT	A_DATE
101	Anil	andheri	7000	01-JAN-06
102	Sunil	virar	5000	15-JUL-06
103	Jay	villeparle	6500	12-MAR-06
104	Vijay	andheri	8000	17-SEP-06
105	Keyur	dadar	7500	19-NOV-06
106	Mayur	borivali	5500	21-DEC-06

Create table borrow (loanno, cname, bname, amount).

Code:

Create table borrow (loanno varchar2(5), cname varchar2(15), bname varchar2(10), amount number(7,2))

Output:

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPOSIT	A_NO	Varchar2	5	-	-	-	✓	-	-
	CNAME	Varchar2	15	-	-	-	✓	-	-
	BNAME	Varchar2	10	-	-	-	✓	-	-
	AMOUNT	Number	-	7	2	-	✓	-	-
	A_DATE	Date	7	-	-	-	✓	-	-
									1 - 5

Perform following queries

(1) Retrieve all data from employee, jobs and deposit.

Query: select *from job

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select *from job
```

Results Explain Describe Saved SQL History

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL
it_prog	programmer	4000	10000
mk_mgr	marketing manager	9000	15000
fi_mgr	finance manager	8200	12000
fi_acc	account	4200	9000
lec	lecturer	6000	17000
comp_op	computer operator	1500	3000

6 rows returned in 0.02 seconds CSV Export

Query: select *from employee

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select *from employee
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
101	smith	800	-	20	shah	machine learning	fi_mgr	toronto	105	09-AUG-96
102	snehal	1600	300	25	gupta	data science	lec	las vegas	-	14-MAR-96
103	adama	1100	0	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95
104	aman	3000	-	15	sharma	virtual reality	comp_op	mexico	12	02-OCT-97
105	anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98
106	sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97
107	anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97

7 rows returned in 0.00 seconds CSV Export

Query: select *from deposit

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select *from deposit
```

Results Explain Describe Saved SQL History

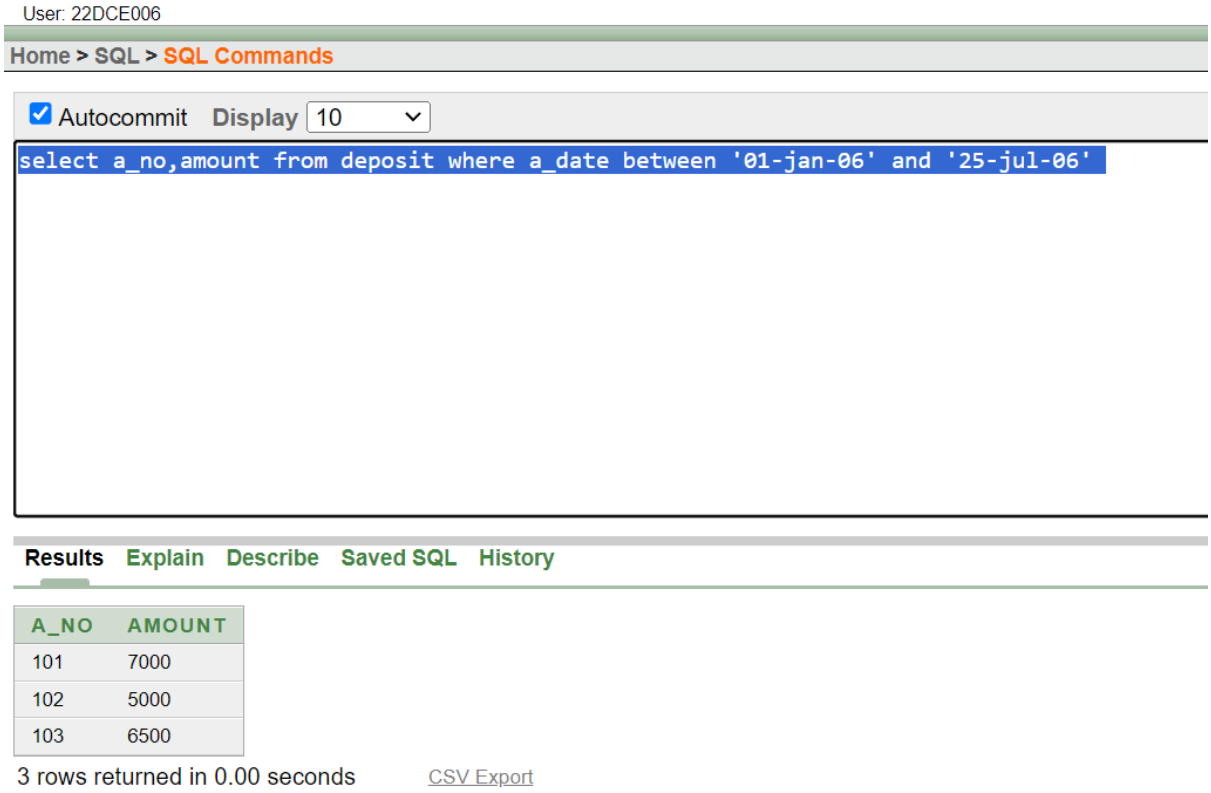
A_NO	CNAME	BNAME	AMOUNT	A_DATE
101	anil	andheri	7000	01-JAN-06
102	sunil	virar	5000	15-JUL-06
103	jay	villepark	6500	12-MAR-06
104	vijay	andheri	8000	17-SEP-06
105	keyur	dadar	7500	19-NOV-06
106	mayur	borivali	5500	21-DEC-06

6 rows returned in 0.00 seconds CSV Export

(2) Give details of account no. and deposited rupees of customers having account opened between dates 01-01-06 and 25-07-06.

Query: select a_no,amount from deposit where a_date between '01-jan-06' and '25-jul-06'

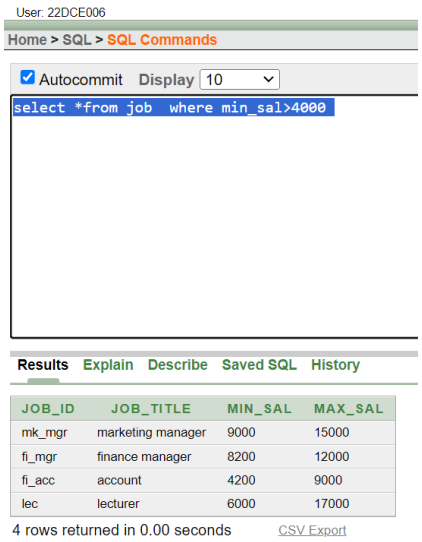
Output:



(3) Display all jobs with minimum salary is greater than 4000.

Query: select *from job where min_sal>4000

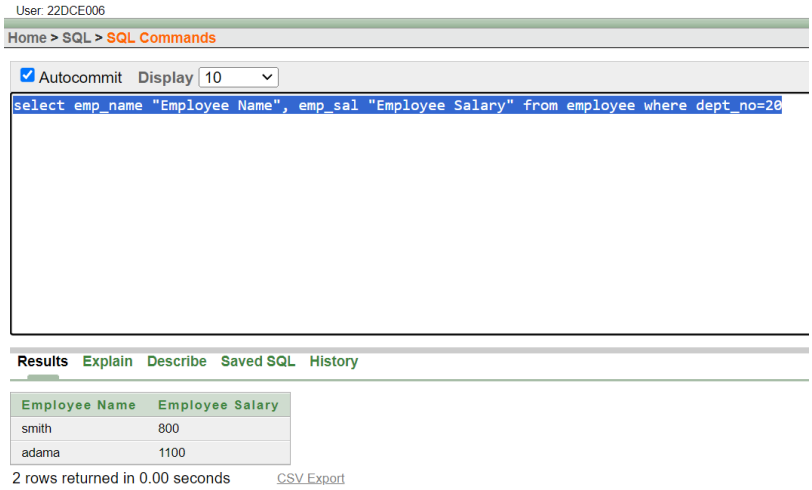
Output:



(4) Display name and salary of employee whose department no is 20. Give alias name to name of employee.

Query: select emp_name "Employee Name", emp_sal "Employee Salary" from employee where dept_no=20

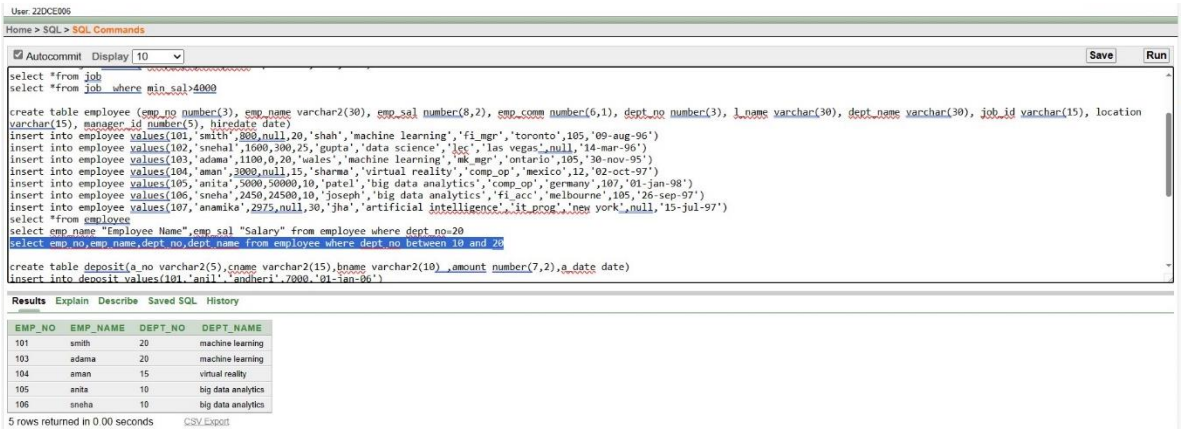
Output:



(5) Display employee no, name and department details of those employee whose department lies in (10,20).

Query: select emp_no,emp_name,dept_no,dept_name from employee where dept_no between 10 and 20

Output:



(6) Display the non-null values of employees.

Query: select *from employee where emp_comm is not null

Output:

User: 22DCE006
Home > SQL > SQL Commands

Autocommit Display 10

```
select *from employee where emp_comm is not null
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
102	snehal	1600	300	25	gupta	data science	lec	las vegas	-	14-MAR-96
103	adama	1100	0	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95
105	anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98
106	sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97

4 rows returned in 0.00 seconds CSV Export

(7) Display name of customer along with its account no (both columns should be displayed as one) whose amount is not equal to 8000 Rs.

Query: select cname ||' '|| a_no "Name and AccountNumber"from deposit where amount!=8000

Output:

User: 22DCE006
Home > SQL > SQL Commands

Autocommit Display 10

```
select cname ||'  '|| a_no "Name and AccountNumber"from deposit where amount!=8000
```

Results Explain Describe Saved SQL History

Name And AccountNumber
anil 101
sunil 102
jay 103
keyur 105
mayur 106

5 rows returned in 0.00 seconds CSV Export

(8) Display the content of job details with minimum salary either 2000 or 4000.

Query: select *from job where min_sal = 2000 or min_sal = 4000

Output:

User: 22DCE006
Home > SQL > SQL Commands

Autocommit Display 10

```
select *from job where min_sal = 2000 or min_sal = 4000
```

Results Explain Describe Saved SQL History

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL
it_prog	programmer	4000	10000

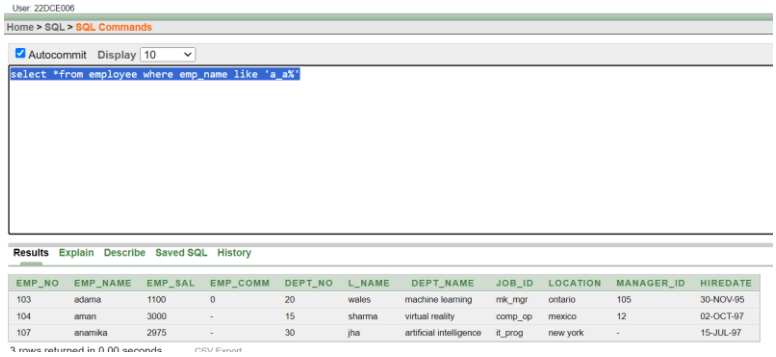
1 rows returned in 0.00 seconds CSV Export

To study various options of LIKE predicate

(1) Display all employee whose name start with ‘A’ and third character is ‘‘a’.

Query: select *from employee where emp_name like 'a_a%'

Output:

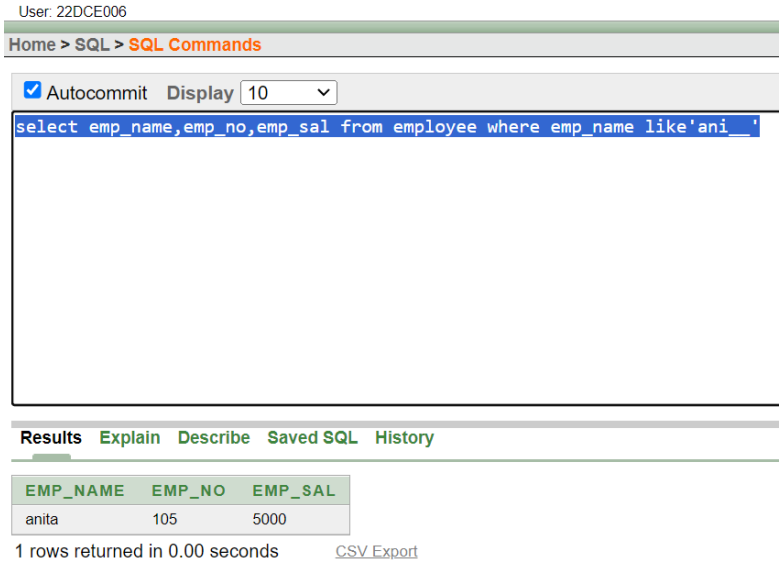


EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
103	adama	1100	0	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95
104	aman	3000	-	15	sharma	virtual reality	comp_op	mexico	12	02-OCT-97
107	anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97

(2) Display name, number and salary of those employees whose name is 5 characters long and first three characters are ‘Ani’.

Query: select emp_name,emp_no,emp_sal from employee where emp_name like'ani__'

Output:



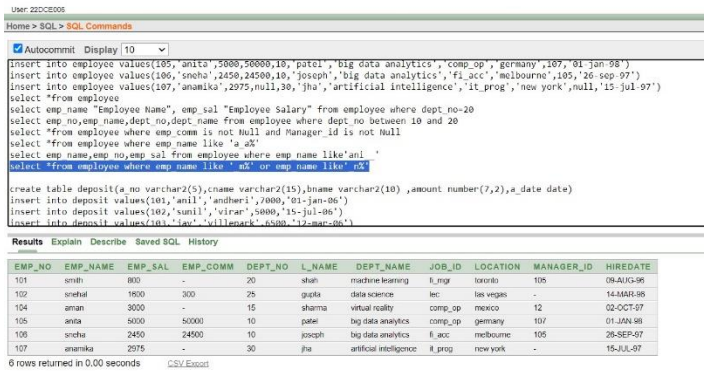
EMP_NAME	EMP_NO	EMP_SAL
anita	105	5000

1 rows returned in 0.00 seconds CSV Export

(3) Display all information of employee whose second character of name is either ‘M’ or ‘N’.

Query: select *from employee where emp_name like '_m%' or emp_name like'_n%'

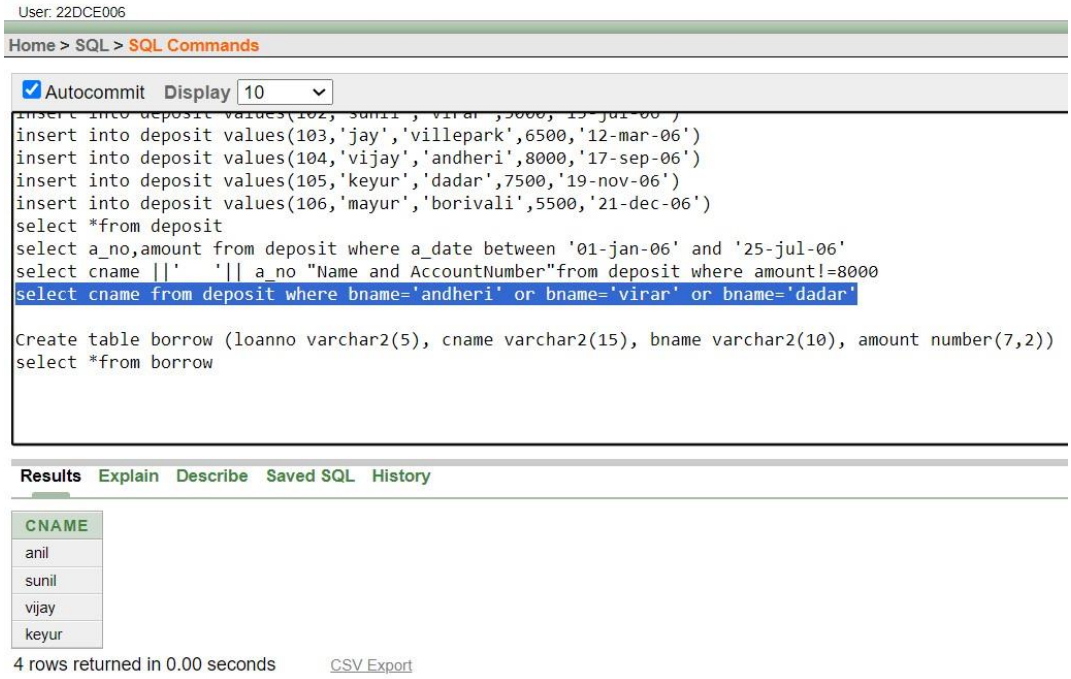
Output:



(4) Find the list of all customer name whose branch is in ‘andheri’ or ‘dadar’ or ‘virar’.

Query: select cname from deposit where bname='andheri' or bname='virar' or bname='dadar'

Output:



(5) Display the job name whose first three character in job id field is ‘FI_’.

Query: select job_title from job where job_id like 'fi$_%' escape '$'

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select job_title from job where job_id like 'fi$_%' escape '$'
```

Results Explain Describe Saved SQL History

JOB_TITLE
finance manager
account

2 rows returned in 0.00 seconds CSV Export

(6) Display the title/name of job who's last three character are '_MGR' and their maximum salary is greater than Rs 12000.

Query: select job_title from job where job_id like '%_mgr' and max_sal>12000

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select job_title from job where job_id like '%_mgr' and max_sal>12000
```

Results Explain Describe Saved SQL History

JOB_TITLE
marketing manager

1 rows returned in 0.00 seconds CSV Export

(7) Display the non-null values of employees and also employee name second character should be 'n' and string should be 5-character long.

Query: select *from employee where emp_comm is not null and emp_name like '_n___'

Output:

User: 22DCE006
Home > SQL > SQL Commands

Autocommit Display 10

```
select *from employee where emp_comm  is not null  and emp_name like '_n__'
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
105	anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98
106	sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97

2 rows returned in 0.00 seconds CSV Export

(8) Display the null values of employee and also employee name's third character should be 'a'.

Query: select *from employee where emp_comm is null and emp_name like '__a%'

Output:

User: 22DCE006
Home > SQL > SQL Commands

Autocommit Display 10

```
select *from employee where emp_comm is null and emp_name like '__a%'
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
104	aman	3000	-	15	sharma	virtual reality	comp_op	mexico	12	02-OCT-97
107	anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97

2 rows returned in 0.00 seconds CSV Export

(9) What will be output if you are giving LIKE predicate as '%_%' ESCAPE '\'

Query: select *from job where job_id like '%_%' escape '\'

Output:

User: 22DCE006
Home > SQL > SQL Commands

Autocommit Display 10

```
select *from job where job_id like '%\_%' escape '\'
```

Results Explain Describe Saved SQL History

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL
it_prog	programmer	4000	10000
mk_mgr	marketing manager	9000	15000
fi_mgr	finance manager	8200	12000
fi_acc	account	4200	9000
comp_op	computer operator	1500	3000

5 rows returned in 0.00 seconds CSV Export

CONCLUSION: From this practical, I learned about the use of like predicate and between keyword. Also displaying non null values with the logic of escape characters.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 3

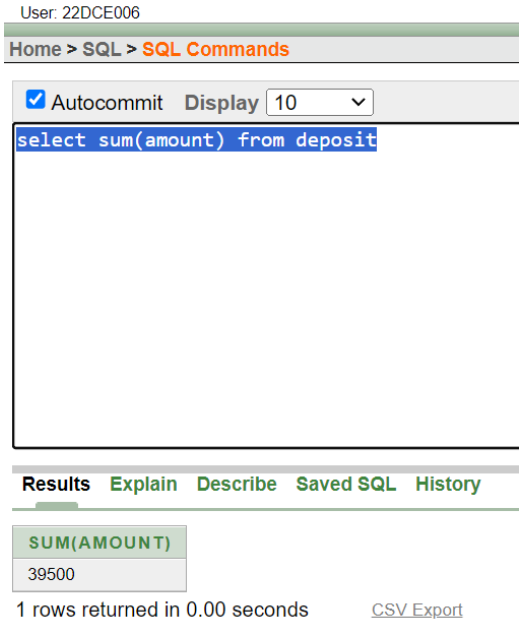
Aim:

To Perform various data manipulation commands, aggregate functions and sorting concept on all created tables.

(1) List total deposit from deposit.

Query: select sum(amount) from deposit

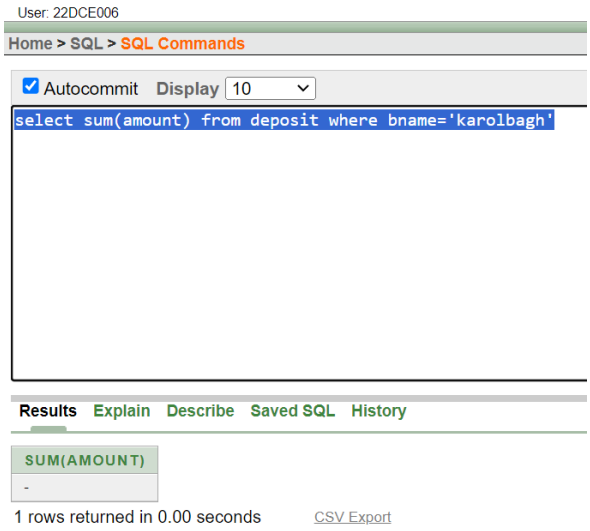
Output:



(2) List total loan from karolbagh branch

Query: select sum(amount) from deposit where bname='karolbagh'

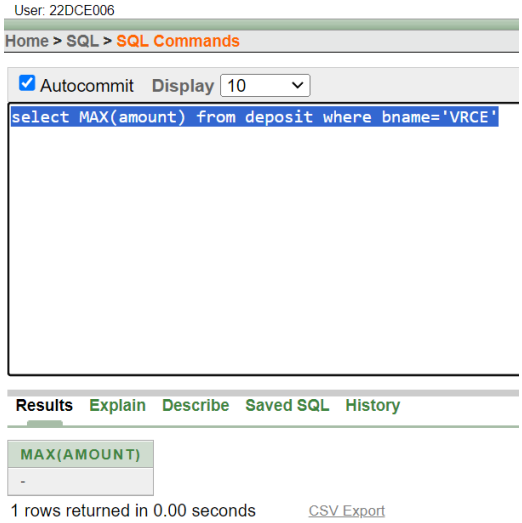
Output:



(3) Give maximum loan from branch vrce.

Query: select MAX(amount) from deposit where bname='VRCE'

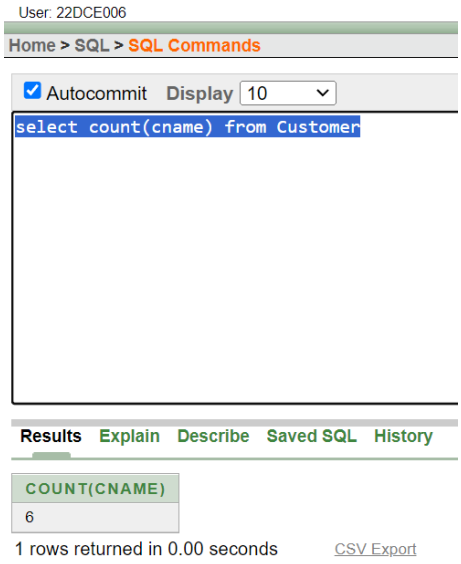
Output:



(4) Count total number of customers

Query: select count(cname) from Customer

Output:



(5) Count total number of customer's cities.

Query: select count(distinct city) from Customer

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select count(distinct CITY) from Customer
```

Results Explain Describe Saved SQL History

COUNT(DISTINCTCITY)
4

1 rows returned in 0.00 seconds CSV Export

(6) Create table supplier from employee with all the columns.

Query: create table supplier as (select * from employee)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
create table supplier as (select * from employee)
```

Results Explain Describe Saved SQL History

```
Table created.
```

0.02 seconds

(7) Create table sup1 from employee with first two columns.

Query: create table sup1 as (select emp_no,emp_name from employee)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
create table sup1 as (select emp_no,emp_name from employee)
```

Results Explain Describe Saved SQL History

```
Table created.
```

0.00 seconds

(8) Create table sup2 from employee with no data

Query: create table sup2 as (select * from employee where 1=0)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
create table sup2 as (select * from employee where 1=0)
```

Results Explain Describe Saved SQL History

```
Table created.
```

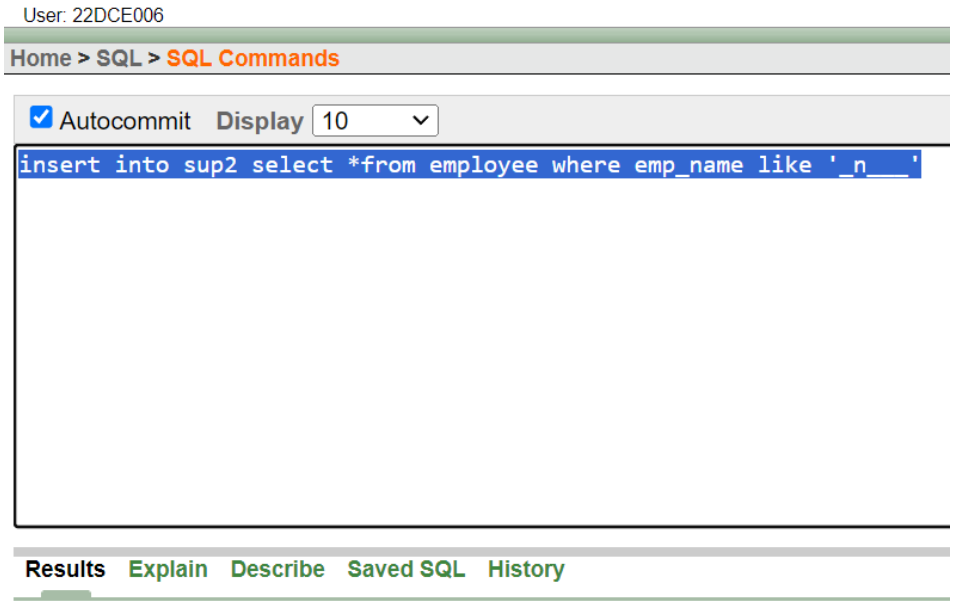
0.01 seconds

(9) Insert the data into sup2 from employee whose second character should be ‘n’ and string should be 5 characters long

in employee name field.

Query: insert into sup2 select *from employee where emp_name like '_n___'

Output:



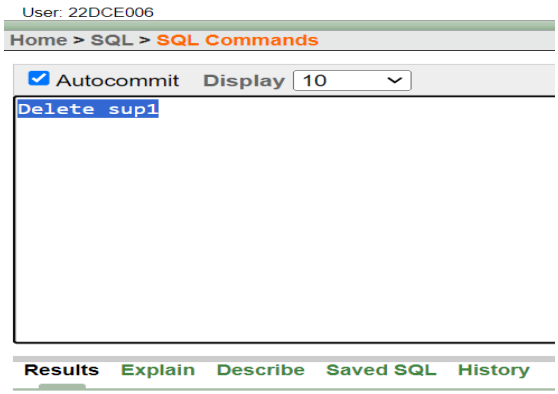

```
2 row(s) inserted.
```

0.00 seconds

(10) Delete all the rows from sup1.

Query: delete sup1

Output:



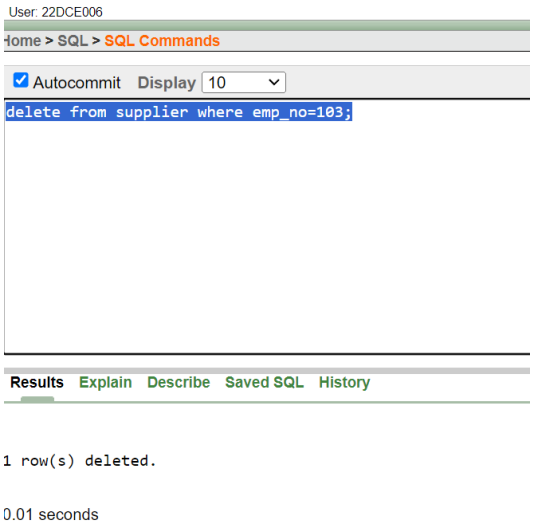

```
7 row(s) deleted.
```

0.02 seconds

(11) Delete the detail of supplier whose sup_no is 103.

Query: delete from supplier where emp_no=103;

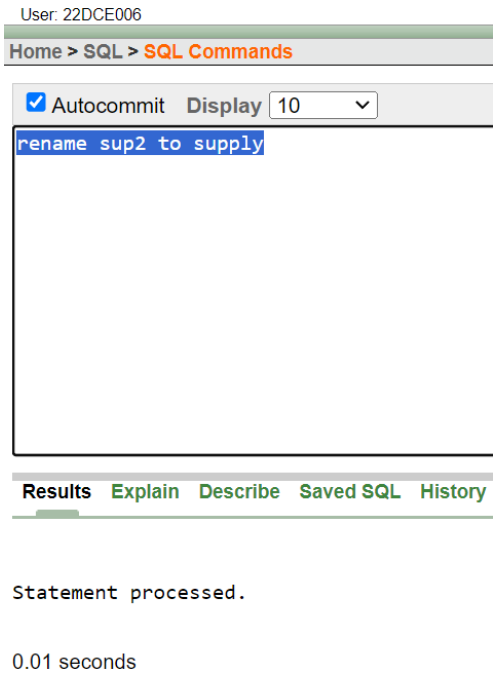
Output:



(12) Rename the table sup2.

Query: rename sup2 to supply

Output:



(13) Destroy table sup1 with all the data.

Query: drop table sup1;

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
drop table sup1;
```

Results Explain Describe Saved SQL History

```
Table dropped.
```

0.05 seconds

(14) Update the value dept_no to 10 where second character of emp. name is ‘m’.

Query: update employee set dept_no=10 where emp_name like '_m%'

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
update employee set dept_no=10 where emp_name like '_m%'
```

Results Explain Describe Saved SQL History

```
2 row(s) updated.
```

0.00 seconds

(15) Update the value of employee name whose employee number is 103.

Query: update employee set emp_name='David' where emp_no = 103

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
update employee set emp_name='David' where emp_no = 103
```

Results Explain Describe Saved SQL History

```
1 row(s) updated.
```

0.00 seconds

(16) Add one column phone to employee with size of column is 10.

Query: alter table employee add phone number(10)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
alter table employee add phone number(10)
```

Results Explain Describe Saved SQL History

```
Table altered.
```

0.03 seconds

(17) Modify the column emp_name to hold maximum of 30 characters.

Query: alter table employee modify emp_name varchar2(30);

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
alter table employee modify emp_name varchar2(30);
```

Results Explain Describe Saved SQL History

```
Table altered.
```

0.02 seconds

(18) Count the total no as well as distinct rows in dept_no column with a condition of salary greater than 1000 of employee

Query: select count(distinct emp_no) from employee where emp_sal>1000

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select count(distinct emp_no) from employee where emp_sal>1000
```

Results Explain Describe Saved SQL History

COUNT(DISTINCTEMP_NO)
6

1 rows returned in 0.00 seconds CSV Export

(19) Display the detail of all employees in ascending order, descending order of their name and no.

Query: select *from employee order by emp_name asc,emp_no desc

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select *from employee order by emp_name asc,emp_no desc
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE	PHONE
103	David	1100	0	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95	-
104	aman	3000	-	10	sharma	virtual reality	comp_op	mexico	12	02-OCT-97	-
107	anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97	-
105	anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98	-
101	smith	800	-	10	shah	machine learning	fi_mgr	toronto	105	09-AUG-96	-
106	sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97	-
102	snehal	1600	300	25	gupta	data science	lec	las vegas	-	14-MAR-96	-

7 rows returned in 0.00 seconds CSV Export

(20) Display the dept_no in ascending order and accordingly display emp_comm in descending order.

Query: select *from employee order by dept_no asc,emp_comm desc

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select *from employee order by dept_no asc,emp_comm desc;
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE	PHONE
101	smith	800	-	10	shah	machine learning	fi_mgr	toronto	105	09-AUG-96	-
104	aman	3000	-	10	sharma	virtual reality	comp_op	mexico	12	02-OCT-97	-
105	anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98	-
106	sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97	-
103	David	1100	0	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95	-
102	snehal	1600	300	25	gupta	data science	lec	las vegas	-	14-MAR-96	-
107	anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97	-

7 rows returned in 0.00 seconds CSV Export

(21) Update the value of emp_comm to 500 where dept_no is 20.

Query: update employee set emp_comm=500 where dept_no = 20

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
update employee set emp_comm=500 where dept_no = 20
```

Results Explain Describe Saved SQL History

```
1 row(s) updated.
```

0.02 seconds

(22) Display the emp_comm in ascending order with null value first and accordingly sort employee salary in descending order.

Query: select *from employee order by emp_comm asc nulls first,emp_sal desc

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select *from employee order by emp_comm asc nulls first,emp_sal desc
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE	PHONE
104	aman	3000	-	10	sharma	virtual reality	comp_op	mexico	12	02-OCT-97	-
107	anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97	-
101	smith	800	-	10	shah	machine learning	fi_mgr	toronto	105	09-AUG-96	-
102	snehal	1600	300	25	gupta	data science	lec	las vegas	-	14-MAR-96	-
103	David	1100	500	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95	-
106	sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97	-
105	anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98	-

7 rows returned in 0.00 seconds CSV Export

(23) Display the emp_comm in ascending order with null value last and accordingly sort emp_no in descending order.

Query: select *from employee order by emp_comm asc nulls last,emp_no desc;

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select *from employee order by emp_comm asc nulls last,emp_no desc;
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE	PHONE
102	snehal	1600	300	25	gupta	data science	lec	las vegas	-	14-MAR-96	-
103	David	1100	500	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95	-
106	sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97	-
105	anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98	-
107	anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97	-
104	aman	3000	-	10	sharma	virtual reality	comp_op	mexico	12	02-OCT-97	-
101	smith	800	-	10	shah	machine learning	fi_mgr	toronto	105	09-AUG-96	-

7 rows returned in 0.00 seconds CSV Export

Conclusion: From this practical I learned about different SQL commands that can be used for data manipulation and also used for sorting the data according to the requirement.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 4

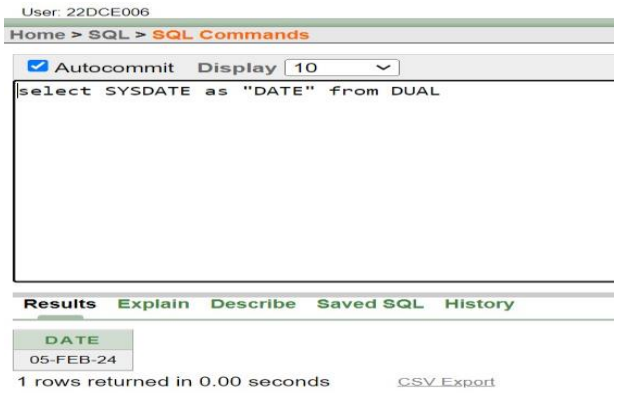
Aim:

To Implement Single-row functions.

(1) Write a query to display the current date. Label the column Date

Query: SELECT SYSDATE as "DATE" FROM dual

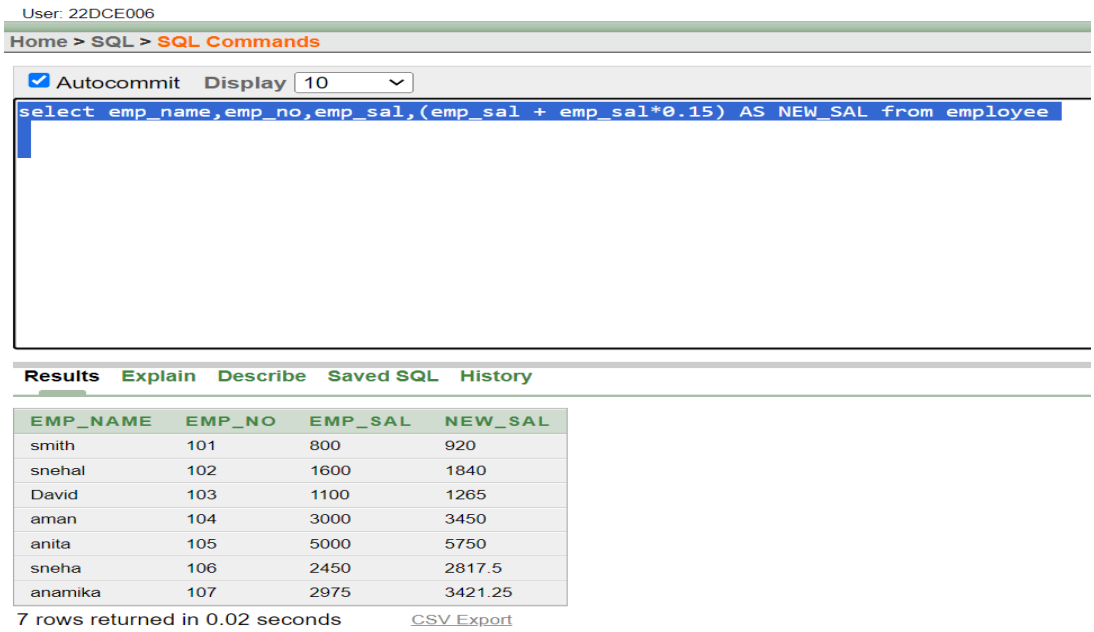
Output:



(2) For each employee, display the employee number, salary, and salary increased by 15% and expressed as a whole number. Label the column New Salary

Query: select emp_name,emp_no,emp_sal,(emp_sal + emp_sal*0.15) AS NEW_SAL from employee

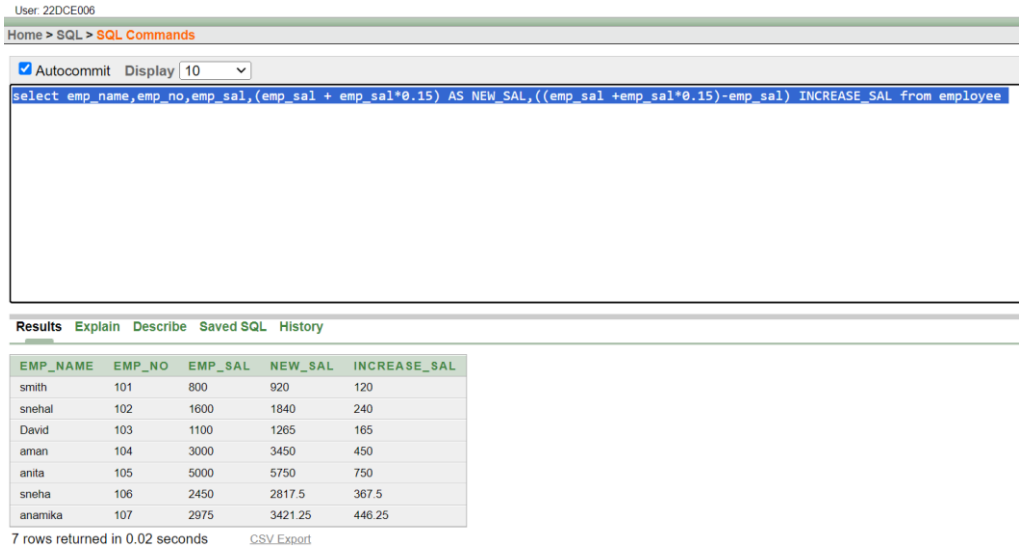
Output:



(3) Modify your query no (2) to add a column that subtracts the old salary from the new salary. Label the column Increase

Query: select emp_name,emp_no,emp_sal,(emp_sal + emp_sal*0.15) AS NEW_SAL,((emp_sal +emp_sal*0.15)-emp_sal) INCREASE_SAL from employee

Output:

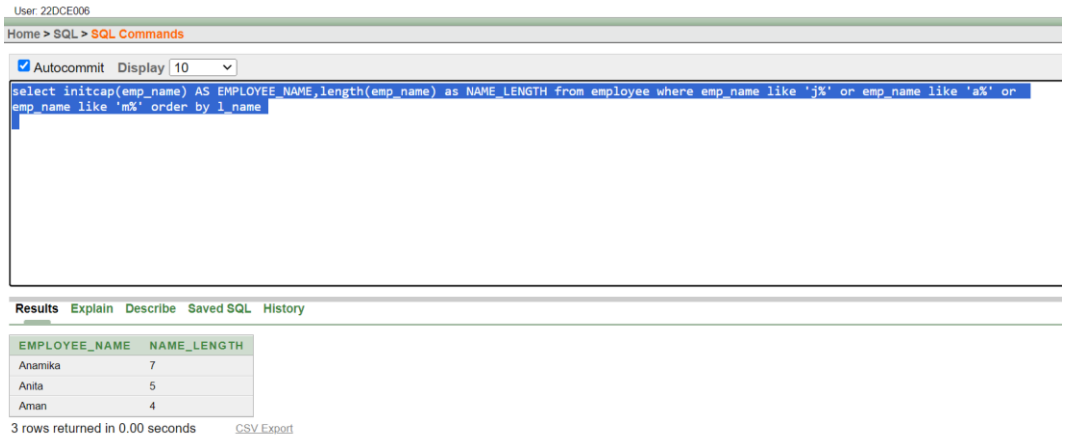


EMP_NAME	EMP_NO	EMP_SAL	NEW_SAL	INCREASE_SAL
smith	101	800	920	120
snehal	102	1600	1840	240
David	103	1100	1265	165
aman	104	3000	3450	450
anita	105	5000	5750	750
sneha	106	2450	2817.5	367.5
anamika	107	2975	3421.25	446.25

(4) Write a query that displays the employee's names with the first letter capitalized and all other letters lowercase, and the length of the names, for all employees whose name starts with J, A, or M. Give each column an appropriate label. Sort the results by the employees' last names.

Query: select initcap(emp_name) AS EMPLOYEE_NAME,length(emp_name) as NAME_LENGTH from employee where emp_name like 'j%' or emp_name like 'a%' or emp_name like 'm%' order by l_name

Output:



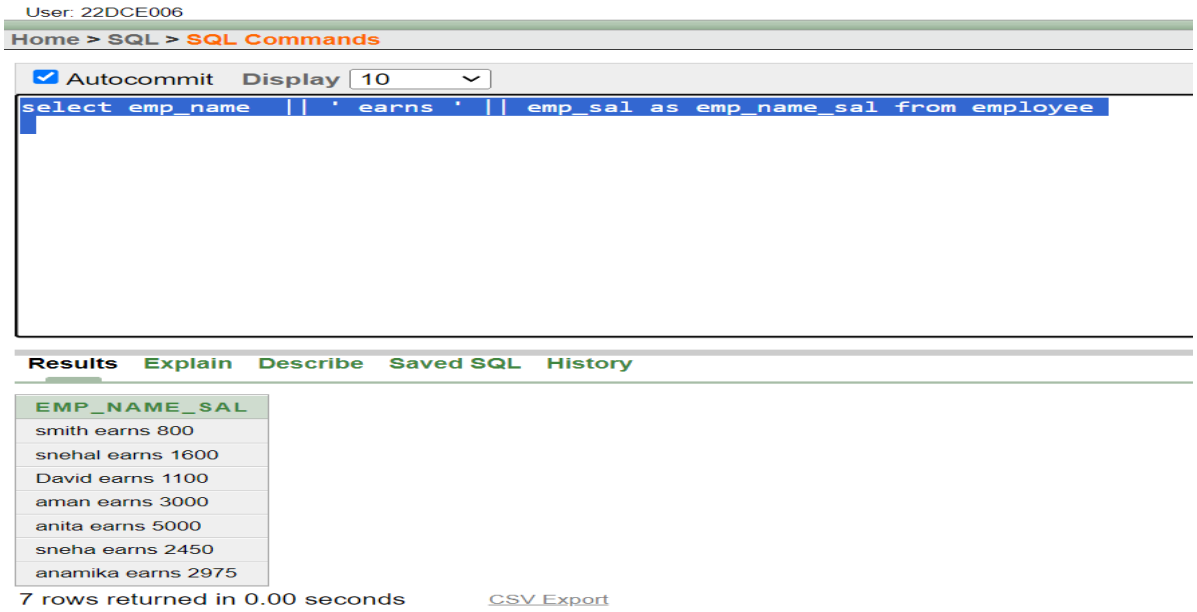
EMPLOYEE_NAME	NAME_LENGTH
Anamika	7
Anita	5
Aman	4

(5) Write a query that produces the following for each employee:

<employee last name> earns <salary> monthly

Query: select emp_name || ' earns ' || emp_sal as emp_name_sal from employee

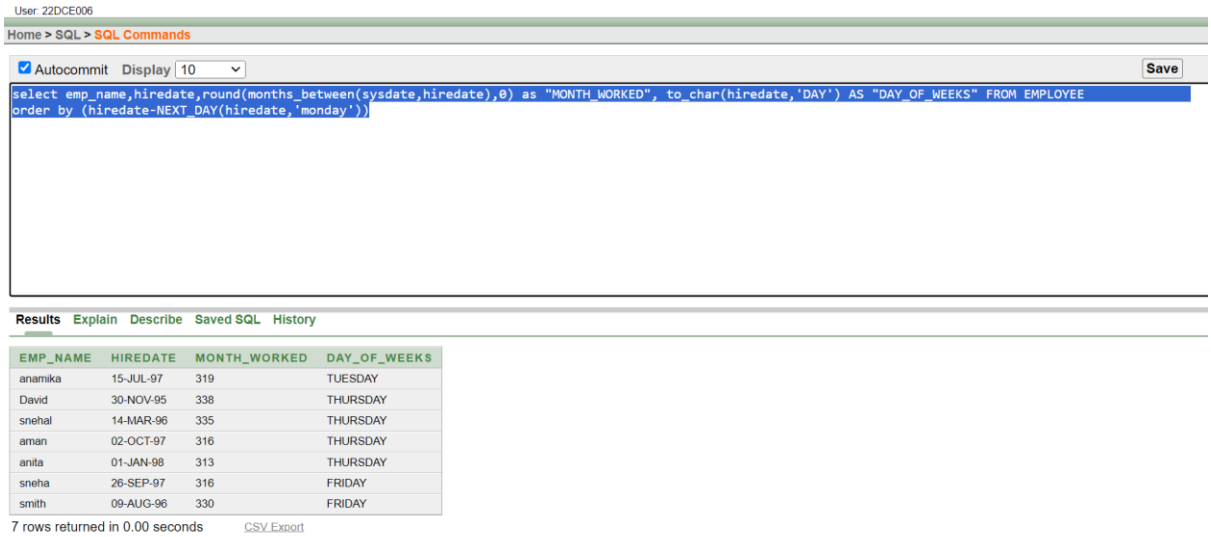
Output:



(6) Display the name, date, number of months employed and day of the week on which the employee has started. Order the results by the day of the week starting with Monday.

Query: select emp_name,hiredate,round(months_between(sysdate,hiredate),0) as "MONTH_WORKED", to_char(hiredate,'DAY') AS "DAY_OF_WEEKS" from employee order by (hiredate-NEXT_DAY(hiredate,'monday'))

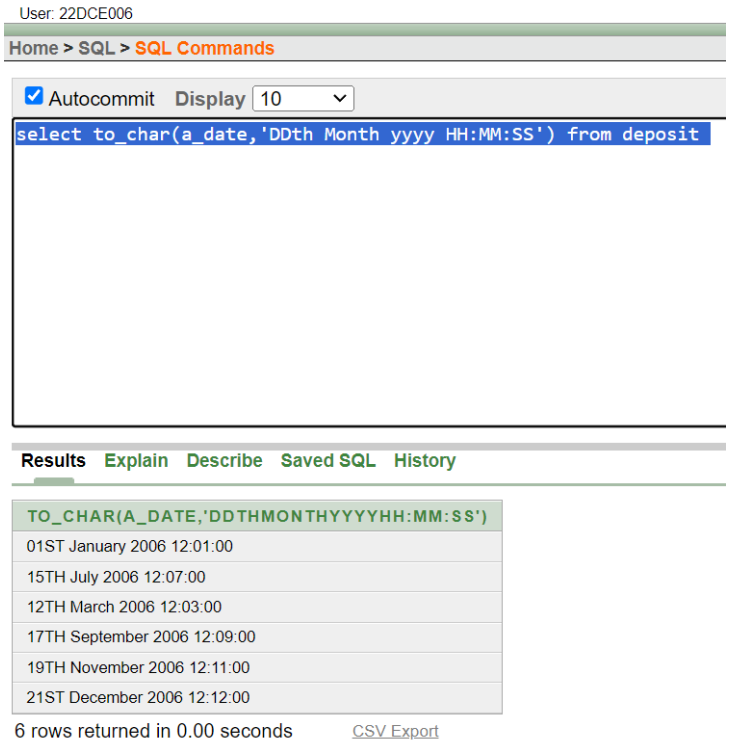
Output:



(7) Display the date of emp in a format that appears as Seventh of June 1994 12:00:00 AM.

Query: select to_char(a_date,'DDth Month yyyy HH:MM:SS') from deposit

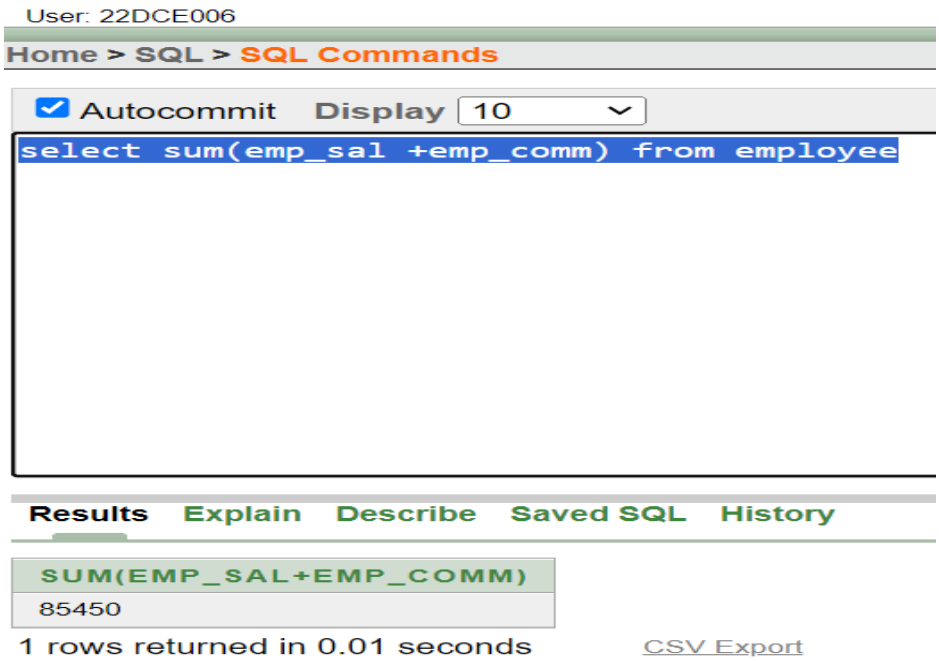
Output:



(8) Write a query to calculate the annual compensation of all employees (sal +comm.).

Query: select sum(emp_sal +emp_comm) from employee

Output:



Conclusion: From this practical I learned about different SQL commands that can be used for data manipulation and single row SQL functions which are useful to getting one output per row.

Staff Signature:

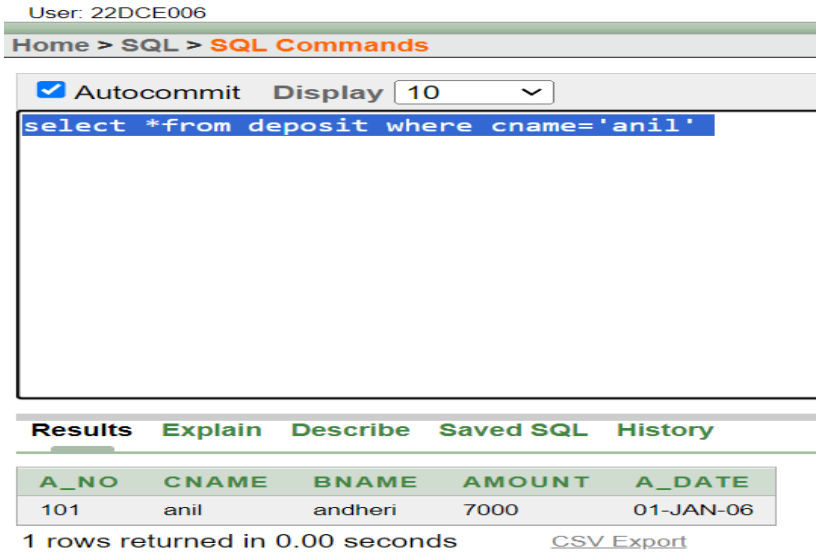
Grade:

Remarks by the Staff:

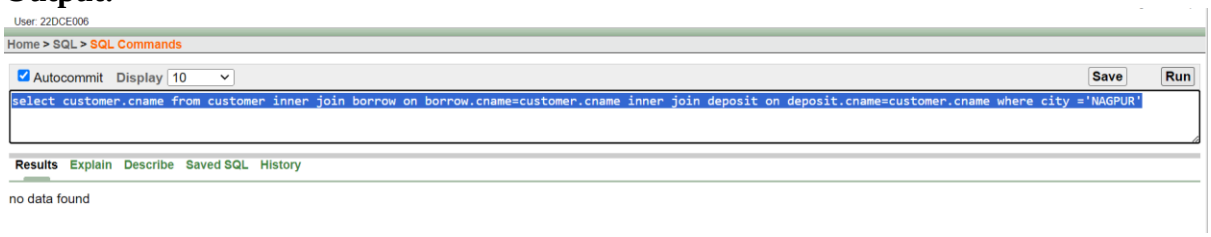
Practical 5

Aim: Displaying data from Multiple Tables (join)

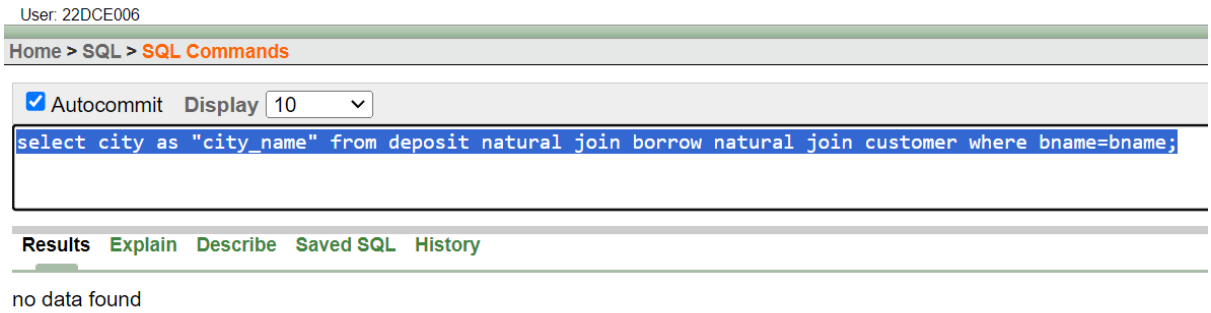
(1) Give details of customers ANIL.
Query : select *from deposit where cname='anil'
Output:



(2) Give name of customer who are borrowers and depositors and having living city nagpur
Query : select customer.cname from customer inner join borrow on borrow.cname=customer.cname inner join deposit on deposit.cname=customer.cname where city ='NAGPUR'
Output:



(3) Give city as their city name of customers having same living branch.
Query : select city as "city_name" from deposit natural join borrow natural join customer where bname=bname;
Output:



(4) Write a query to display the last name, department number, and department name for all employees.

Query : select l_name,dept_no,dept_name from employee

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select l_name,dept_no,dept_name from employee
```

Results Explain Describe Saved SQL History

L_NAME	DEPT_NO	DEPT_NAME
shah	10	machine learning
gupta	25	data science
wales	20	machine learning
sharma	10	virtual reality
patel	10	big data analytics
joseph	10	big data analytics
jha	30	artificial intelligence

7 rows returned in 0.00 seconds CSV Export

(5) Create a unique listing of all jobs that are in department 30. Include the location of the department in the output

Query : select job.job_id from job inner join employee on employee.job_id=job.job_id and dept_no=30

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select job.job_id from job inner join employee on employee.job_id=job.job_id and dept_no=30;
```

Results Explain Describe Saved SQL History

JOB_ID
it_prog

1 rows returned in 0.02 seconds CSV Export

(6) Write a query to display the employee's name, department number, and department name for all employees who work in NEW YORK.

Query : select emp_name,dept_no,dept_name from employee where location='new york'

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select emp_name,dept_no,dept_name from employee where location='new york'
```

Results Explain Describe Saved SQL History

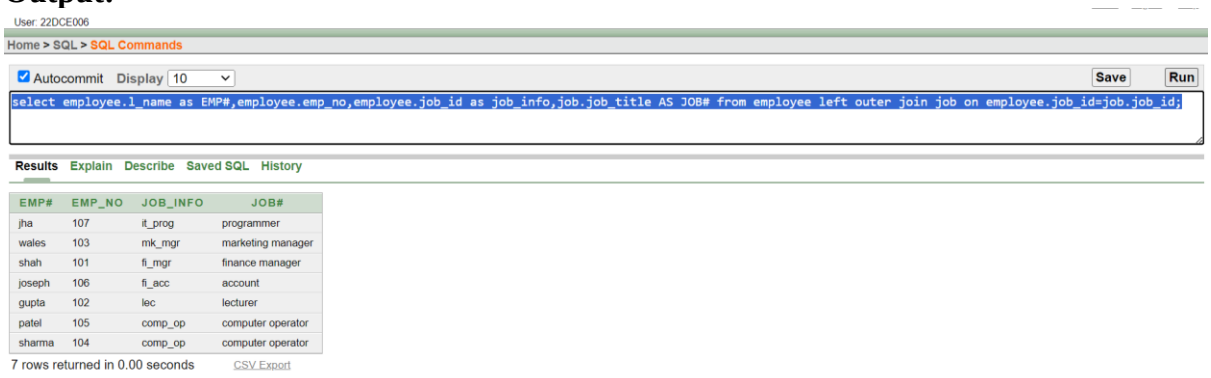
EMP_NAME	DEPT_NO	DEPT_NAME
anamika	30	artificial intelligence

1 rows returned in 0.00 seconds CSV Export

(7) Display the employee's last name and employee number along with their manager's last name and manager number. Label the columns Employee, Emp#, Manager, and Mgr#, respectively.

Query : select employee.l_name as EMP#,employee.emp_no,employee.job_id as job_info,job.job_title AS JOB# from employee left outer join job on employee.job_id=job.job_id;

Output:

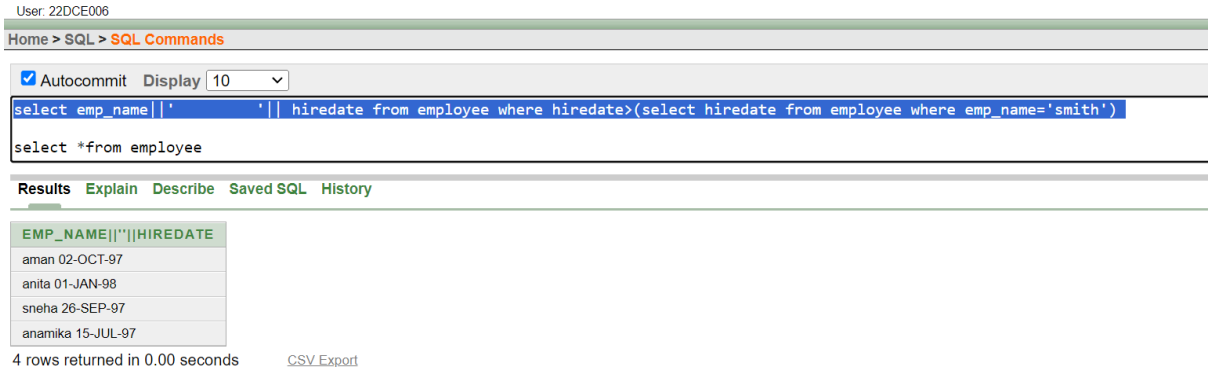


EMP#	EMP_NO	JOB_INFO	JOB#
jha	107	it_prog	programmer
wales	103	mk_mgr	marketing manager
shah	101	fi_mgr	finance manager
joseph	106	fi_acc	account
gupta	102	lec	lecturer
patel	105	comp_op	computer operator
sharma	104	comp_op	computer operator

(8) Create a query to display the name and hire date of any employee hired after employee "smith".

Query : select emp_name||' '|| hiredate from employee where hiredate>(select hiredate from employee where emp_name='smith')

Output:



EMP_NAME||''||HIREDATE
aman 02-OCT-97
anita 01-JAN-98
sneha 26-SEP-97
anamika 15-JUL-97

Conclusion: From this practical I learned about different SQL commands that can be used for data manipulation and join SQL functions which are useful for getting output by combining multiple tables.

Staff Signature:

Grade:

Remarks by the Staff:

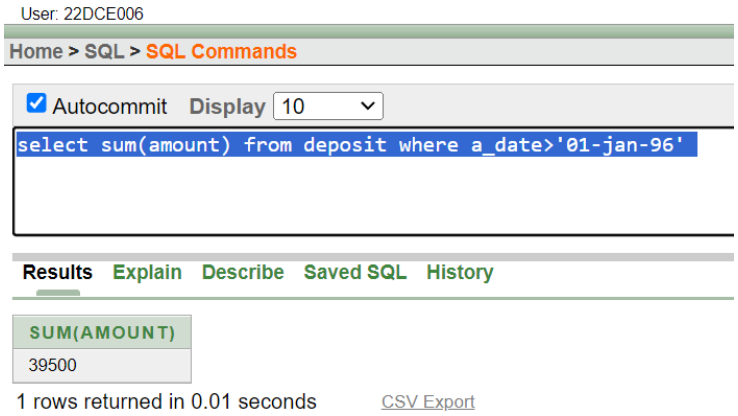
Practical 6

Aim: To apply the concept of Aggregating Data using Group functions.

(1)List total deposit of customer having account date after 1-jan-96.

Query: select sum(amount) from deposit where a_date>'01-jan-96'

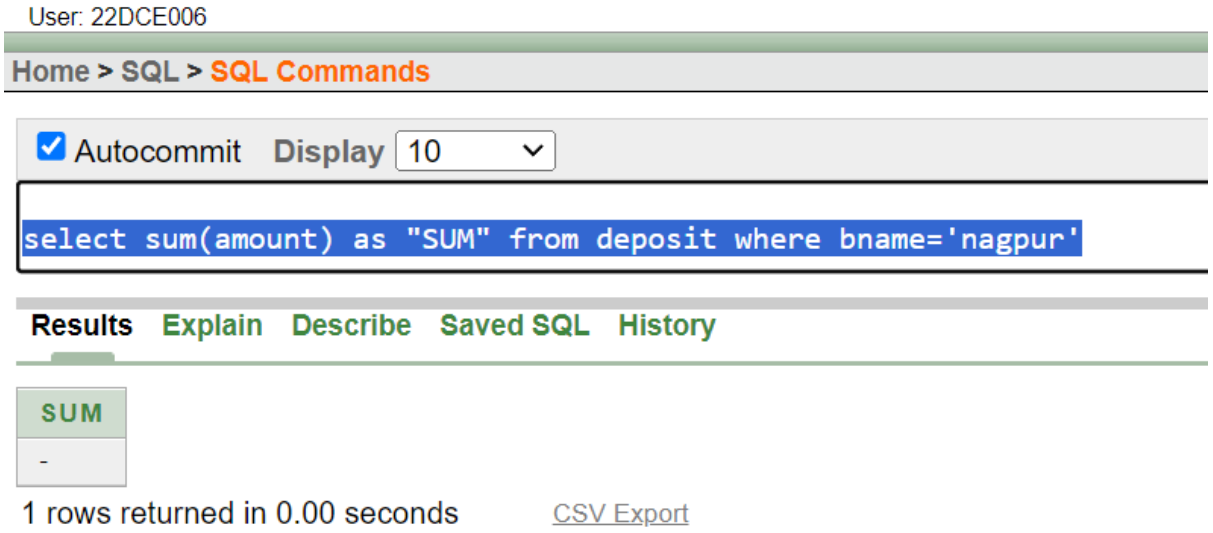
Output:



(2) List total deposit of customers living in city Nagpur.

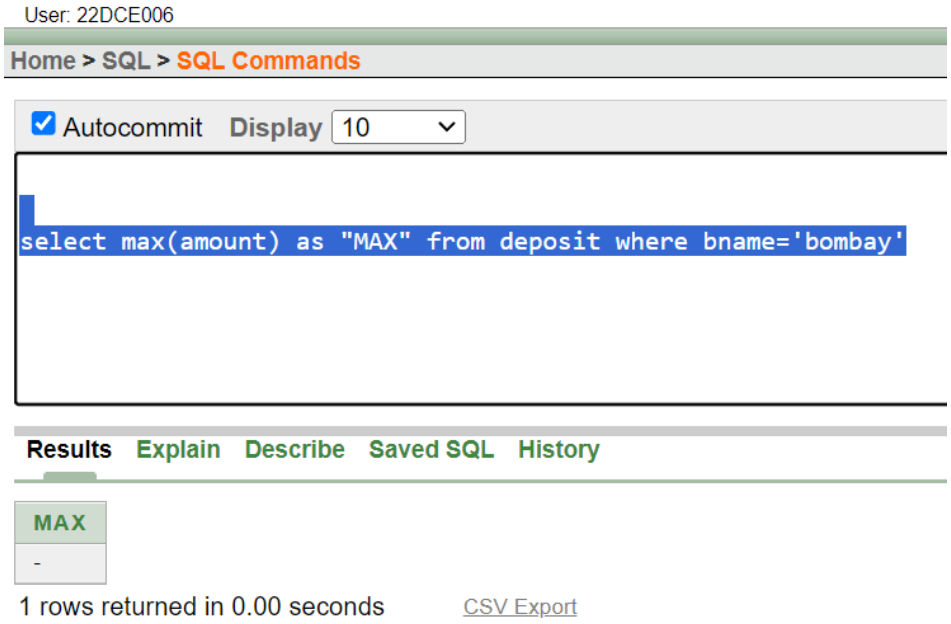
Query: select sum(amount) as "SUM" from deposit where bname='nagpur'

Output:



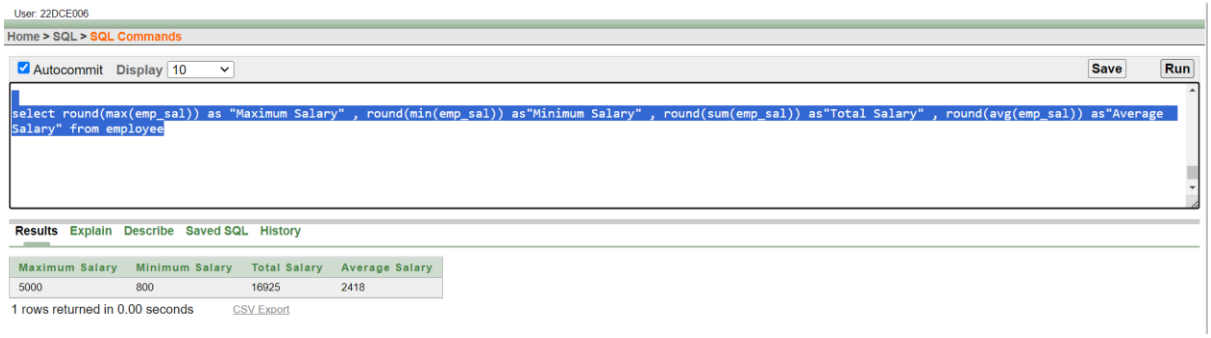
(3) List maximum deposit of customers living in bombay.
Query:select max(amount) as "MAX" from deposit where bname='bombay'

Output:



(4) Display the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number.
Query: select round(max(emp_sal)) as "Maximum Salary" , round(min(emp_sal)) as"Minimum Salary" , round(sum(emp_sal)) as"Total Salary" , round(avg(emp_sal)) as"Average Salary" from employee

Output:



(5) Write a query that displays the difference between the highest and lowest salaries. Label the column DIFFERENCE.

Query: select round(max_sal) as "Highest Salary" , round(min_sal) as"Lowest Salary" , round((max_sal)-(min_sal))as "Difference" from job

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select round(max_sal) as "Highest Salary" , round(min_sal) as"Lowest Salary" , round((max_sal)-(min_sal))as "Difference" from job
```

Results Explain Describe Saved SQL History

Highest Salary	Lowest Salary	Difference
10000	4000	6000
15000	9000	6000
12000	8200	3800
9000	4200	4800
17000	6000	11000
3000	1500	1500

6 rows returned in 0.01 seconds CSV Export

(6) Create a query that will display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998

Query: select count(emp_name) as "Count" from employee where to_char(hiredate,'yy') in('95','96','97','98')

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select count(emp_name) as "Count" from employee where to_char(hiredate,'yy') in('95','96','97','98')
```

Results Explain Describe Saved SQL History

Count
7

1 rows returned in 0.00 seconds CSV Export

(7) Find the average salaries for each department without displaying the respective department numbers.

Query: select dept_no,ROUND(AVG(emp_sal)) from employee GROUP BY dept_no

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT DEPT_NO,ROUND(AVG(emp_sal)) FROM EMPLOYEE GROUP BY DEPT_NO
```

Results Explain Describe Saved SQL History

DEPT_NO	ROUND(AVG(EMP_SAL))
25	1600
30	2975
20	1100
10	2813

4 rows returned in 0.01 seconds CSV Export

(8) Write a query to display the total salary being paid to each job title, within each department.

Query: select dept_name, SUM(emp_sal) from employee GROUP BY dept_name

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select dept_name, SUM(emp_sal) from employee GROUP BY dept_name
```

Results Explain Describe Saved SQL History

DEPT_NAME	SUM(EMP_SAL)
big data analytics	7450
artificial intelligence	2975
machine learning	1900
virtual reality	3000
data science	1600

5 rows returned in 0.00 seconds CSV Export

(9) Find the average salaries > 2000 for each department without displaying the respective department numbers.

Query: select ROUND(AVG(emp_sal)) from employee GROUP BY dept_no HAVING AVG(EMP_SAL)>2000

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select ROUND(AVG(emp_sal)) from employee GROUP BY dept_no HAVING AVG(EMP_SAL)>2000
```

Results Explain Describe Saved SQL History

ROUND(AVG(EMP_SAL))
2975
2813

2 rows returned in 0.00 seconds CSV Export

(10) Display the job and total salary for each job with a total salary amount exceeding 3000 and sorts the list by the total salary.

Query: select job_id,SUM(emp_sal+3000)"EXTENDED SALARY " from employee GROUP BY job_id order by SUM(emp_sal+3000)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select job_id,SUM(emp_sal+3000)"EXTENDED SALARY " from employee GROUP BY job_id order by SUM(emp_sal+3000)
```

Results Explain Describe Saved SQL History

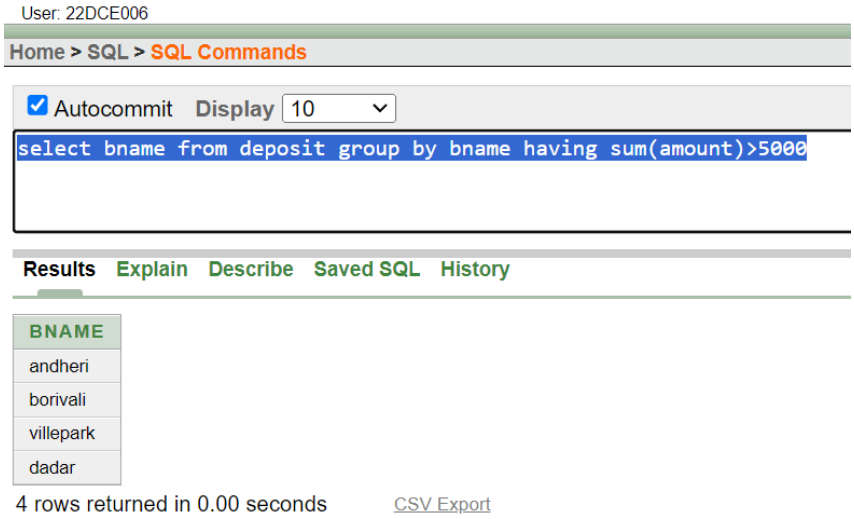
JOB_ID	EXTENDED SALARY
fi_mgr	3800
mk_mgr	4100
lec	4600
fi_acc	5450
it_prog	5975
comp_op	14000

6 rows returned in 0.01 seconds CSV Export

(11) List the branches having sum of deposit more than 5000 and located in city bombay.

Query: select bname from deposit group by bname having sum(amount)>5000

Output:



Conclusion: From this practical I learned about different aggregate functions and other SQL group functions.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 7

Aim: To solve queries using the concept of sub query.

(1) Write a query to display the last name and hire date of any employee in the same department as smith. Exclude smith

Query: select l_name,hiredate from employee where dept_name=(select dept_name from employee where emp_name='smith') and emp_name!='smith'

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select l_name,hiredate from employee where dept_name=(select dept_name from employee where emp_name='smith') and emp_name!='smith'
```

Results Explain Describe Saved SQL History

L_NAME	HIREDATE
wales	30-NOV-95

1 rows returned in 0.00 seconds CSV Export

(2) Give name of customers who are depositors having same branch city of mr. sunil.

Query: select cname from deposit where bname=(select bname from deposit where cname='sunil')

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select cname from deposit where bname=(select bname from deposit where cname='sunil')
```

Results Explain Describe Saved SQL History

CNAME
sunil

1 rows returned in 0.00 seconds CSV Export

(3) Give deposit details and loan details of customer in same city where pramod is living.

Query: select *from deposit,borrow where bname=(select bname from deposit where cname='pramod')

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select *from deposit,borrow where bname=(select bname from deposit where cname='pramod')
```

Results Explain Describe Saved SQL History

no data found

(4) Create a query to display the employee numbers and last names of all employees who earn more than the average salary. Sort the results in ascending order of salary.

Query: select emp_no,l_name from employee where emp_sal>(select avg(emp_sal) from employee) order by emp_sal asc

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select emp_no,l_name from employee where emp_sal>(select avg(emp_sal) from employee) order by emp_sal asc
```

Results Explain Describe Saved SQL History

EMP_NO	L_NAME
106	joseph
107	jha
104	sharma
105	patel

4 rows returned in 0.00 seconds CSV Export

(5) Give names of depositors having same living city as mr. anil and having deposit amount greater than 2000

Query: select cname from deposit where bname=(select bname from deposit where cname='anil' and amount>2000);

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select cname from deposit where bname=(select bname from deposit where cname='anil' and amount>2000);
```

Results Explain Describe Saved SQL History

CNAME
anil
vijay

2 rows returned in 0.00 seconds CSV Export

(6) Display the last name and salary of every employee who reports to patel.

Query: select l_name , emp_sal from employee where manager_id=105;

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select l_name , emp_sal from employee where manager_id=105;
```

Results Explain Describe Saved SQL History

L_NAME	EMP_SAL
shah	800
wales	1100
joseph	2450

3 rows returned in 0.00 seconds CSV Export

(7) Display the department number, name, and job for every employee in the accounting department.

Query: select dept_no,emp_name,job_id from employee where job_id=(select job_id from job where upper(job_title)='ACCOUNT')

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select dept_no,emp_name,job_id from employee where job_id=(select job_id from job where upper(job_title)='ACCOUNT')
```

Results Explain Describe Saved SQL History

DEPT_NO	EMP_NAME	JOB_ID
10	sneha	fi_acc

1 rows returned in 0.00 seconds CSV Export

(8) List the name of branch having highest number of depositors.

Query: select bname from deposit group by bname having count(bname)>=(select max(count(bname)) from deposit group by bname)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select bname from deposit group by bname having count(bname)>=(select max(count(bname)) from deposit group by bname)
```

Results Explain Describe Saved SQL History

BNAME
andheri

1 rows returned in 0.02 seconds CSV Export

(9) Give the name of cities where in which the maximum numbers of branches are located.

Query: select bname from deposit group by bname having count(bname)>=(select max(count(bname)) from deposit group by bname)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
select bname from deposit group by bname having count(bname)>=(select max(count(bname)) from deposit group by bname)
```

Results Explain Describe Saved SQL History

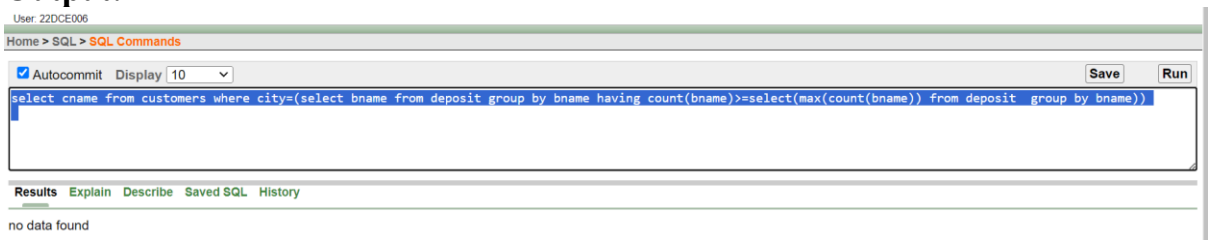
BNAME
andheri

1 rows returned in 0.02 seconds CSV Export

(10) Give name of customers living in same city where maximum depositors are located.

Query: select cname from customers where city=(select bname from deposit group by bname having count(bname)>=select(max(count(bname)) from deposit group by bname))

Output:



Conclusion: From this practical I learned about different aggregate functions , application of sub-queries and other SQL group functions.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 8

Aim: Manipulating Data

(1) Give 10% interest to all depositors.

Query: update deposit set amount=(amount)+(amount*0.1)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
update deposit set amount=(amount)+(amount*0.1)
select *from deposit
```

Results Explain Describe Saved SQL History

A_NO	CNAME	BNAME	AMOUNT	A_DATE
101	anil	andheri	7700	01-JAN-06
102	sunil	virar	5500	15-JUL-06
103	jay	villepark	7150	12-MAR-06
104	vijay	andheri	8800	17-SEP-06
105	keyur	dadar	8250	19-NOV-06
106	mayur	borivali	6050	21-DEC-06

6 rows returned in 0.00 seconds CSV Export

(2) Give 10% interest to all depositors having branch vrce

Query: update deposit set amount=(amount)+(amount*0.1) where bname='vrce'

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
update deposit set amount=(amount)+(amount*0.1) where bname='vrce'
select *from deposit
```

Results Explain Describe Saved SQL History

```
0 row(s) updated.
```

(3) Give 10% interest to all depositors living in nagpur and having branch city bombay.

Query: update deposit set amount=(amount)+(amount*0.1) where bname='nagpur' or bname='bombay'

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
update deposit set amount=(amount)+(amount*0.1) where bname='nagpur' or bname='bombay'
select *from deposit
```

Results Explain Describe Saved SQL History

```
0 row(s) updated.
```

0.00 seconds

(4) Write a query which changes the department number of all employees with empno 7788's job to employee 7844'current department number.

Query: update employee set emp_no=7844 where emp_no=7788

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
update employee set emp_no=7844 where emp_no=7788
```

Results Explain Describe Saved SQL History

```
0 row(s) updated.
```

(5) Transfer 10 Rs from account of anil to sunil if both are having same branch.

Query: update deposit set amount=amount+10 where cname='sunil' and bname in (select bname from deposit where cname='anil')

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
update deposit set amount=amount+10 where cname='sunil' and bname in (select bname from deposit where cname='anil')
```

Results Explain Describe Saved SQL History

```
0 row(s) updated.
```

(6) Give 100 Rs more to all depositors if they are maximum depositors in their respective branch.

Query: UPDATE DEPOSIT SET AMOUNT=AMOUNT+100 WHERE CNAME IN(SELECT CNAME FROM DEPOSIT WHERE AMOUNT = ANY(SELECT MAX(AMOUNT) FROM DEPOSIT GROUP BY BNAME))

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
UPDATE DEPOSIT SET AMOUNT=AMOUNT+100 WHERE CNAME IN(SELECT CNAME FROM DEPOSIT WHERE AMOUNT = ANY(SELECT MAX(AMOUNT) FROM DEPOSIT GROUP BY BNAME))
```

Results Explain Describe Saved SQL History

```
5 row(s) updated.
```

(7) Delete depositors of branches having number of customers between 1 to 3.

Query: DELETE FROM DEPOSIT WHERE BNAME IN(SELECT BNAME FROM DEPOSIT GROUP BY BNAME HAVING COUNT(BNAME)>1 AND COUNT(BNAME)<3)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
DELETE FROM DEPOSIT WHERE BNAME IN(SELECT BNAME FROM DEPOSIT GROUP BY BNAME HAVING COUNT(BNAME)>1 AND COUNT(BNAME)<3)
```

Results Explain Describe Saved SQL History

```
0 row(s) deleted.
```

(8) Delete deposit of vijay.

Query: delete from deposit where cname='vijay'

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
delete from deposit where cname='vijay'
select *from deposit
```

Results Explain Describe Saved SQL History

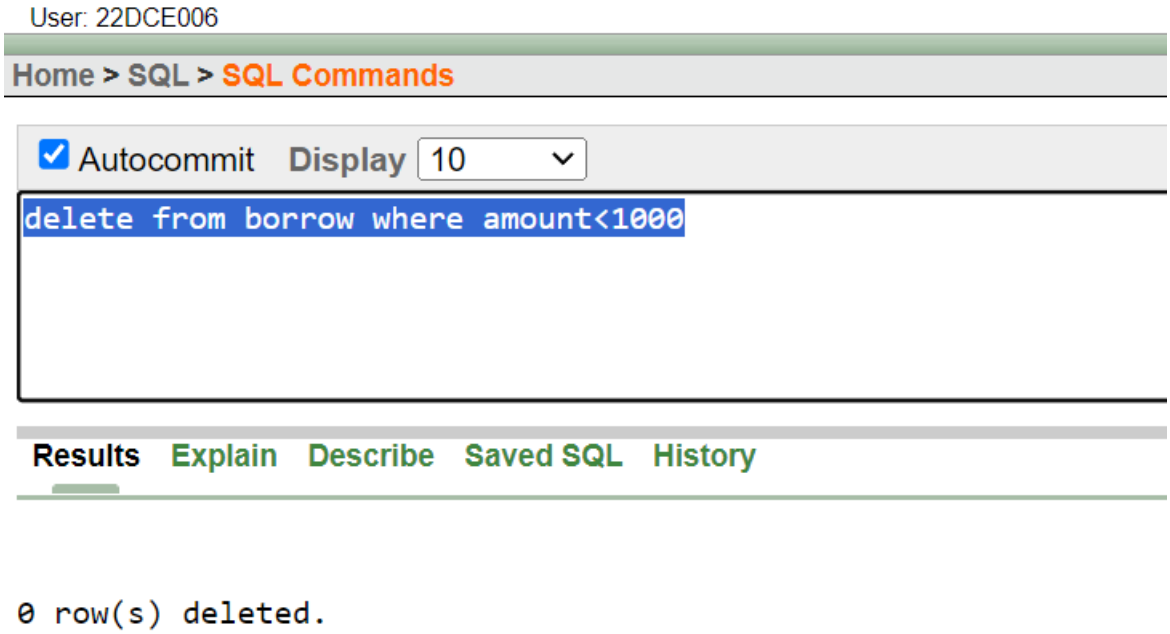
A_NO	CNAME	BNAME	AMOUNT	A_DATE
101	anil	andheri	7700	01-JAN-06
102	sunil	virar	5500	15-JUL-06
103	jay	villepark	7150	12-MAR-06
105	keyur	dadar	8250	19-NOV-06
106	mayur	borivali	6050	21-DEC-06

5 rows returned in 0.00 seconds CSV Export

(9) Delete borrower of branches having average loan less than 1000.

Query: delete from borrow where amount<1000

Output:



Conclusion: From this practical I learned about different functions used for data manipulation and other important queries.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 9

Aim: Add and Remove constraint

(1) Add primary key constraint on job_id in job table.

Query: alter table job add constraint pk_id primary key(job_id)

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
alter table job add constraint pk_id primary key(job_id)
desc job
```

Results Explain Describe Saved SQL History

Object Type TABLE Object JOB

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
JOB	JOB_ID	Varchar2	15	-	-	1	-	-	-
	JOB_TITLE	Varchar2	30	-	-	-	✓	-	-
	MIN_SAL	Number	-	7	2	-	✓	-	-
	MAX_SAL	Number	-	7	2	-	✓	-	-

1 - 4

(2) Add foreign key constraint on employee table referencing job table.

Query: alter table employee add constraint fk_emp foreign key(job_id) references job(job_id)

Output:

Home > SQL > SQL Commands

Autocommit Display 10

```
alter table employee add constraint fk_id foreign key(job_id) references job(job_id);
```

Results Explain Describe Saved SQL History

```
Table altered.
```

(3) Add composite primary key on lock table (lock table does not exist, while creating table add composite key)

Query: create table lock1(s_id number(10),fname varchar2(10) , lname varchar2(10) , primary key(s_id,fname))

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
create table lock1(s_id number(10),fname varchar2(10) , lname varchar2(10) , primary key(s_id,fname))
```

Results Explain Describe Saved SQL History

```
Table created.
```

0.01 seconds

(4) Remove primary key constraint on job_id
Query: alter table employee drop constraint fk_id
Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
alter table employee drop constraint fk_id
```

Results Explain Describe Saved SQL History

```
Table dropped.
```

0.01 seconds

(5) Remove foreign key constraint on employee table
Query: alter table job drop constraint pk_id
Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
alter table job drop constraint pk_id
```

Results Explain Describe Saved SQL History

```
Table dropped.
```

0.04 seconds

Conclusion: From this practical I learned about adding and removing primary key and foreign keys.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 10

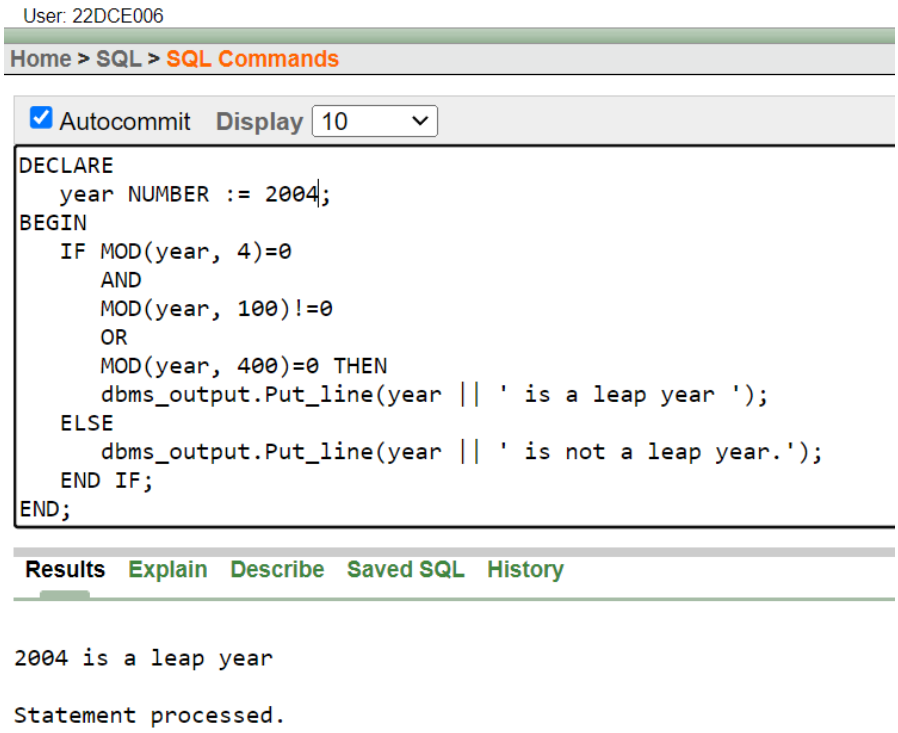
Aim: To perform basic PL/SQL blocks

1. Write a PL-SQL block for checking weather a given year is a Leap year or not.

Query:

```
DECLARE
  year NUMBER := 2004;
BEGIN
  IF MOD(year, 4)=0
    AND
    MOD(year, 100)!=0
    OR
    MOD(year, 400)=0 THEN
    dbms_output.Put_line(year || ' is a leap year ');
  ELSE
    dbms_output.Put_line(year || ' is not a leap year.');
  END IF;
END;
```

Output:



2. Find out whether given string is palindrome or not using for a while and simple loop.

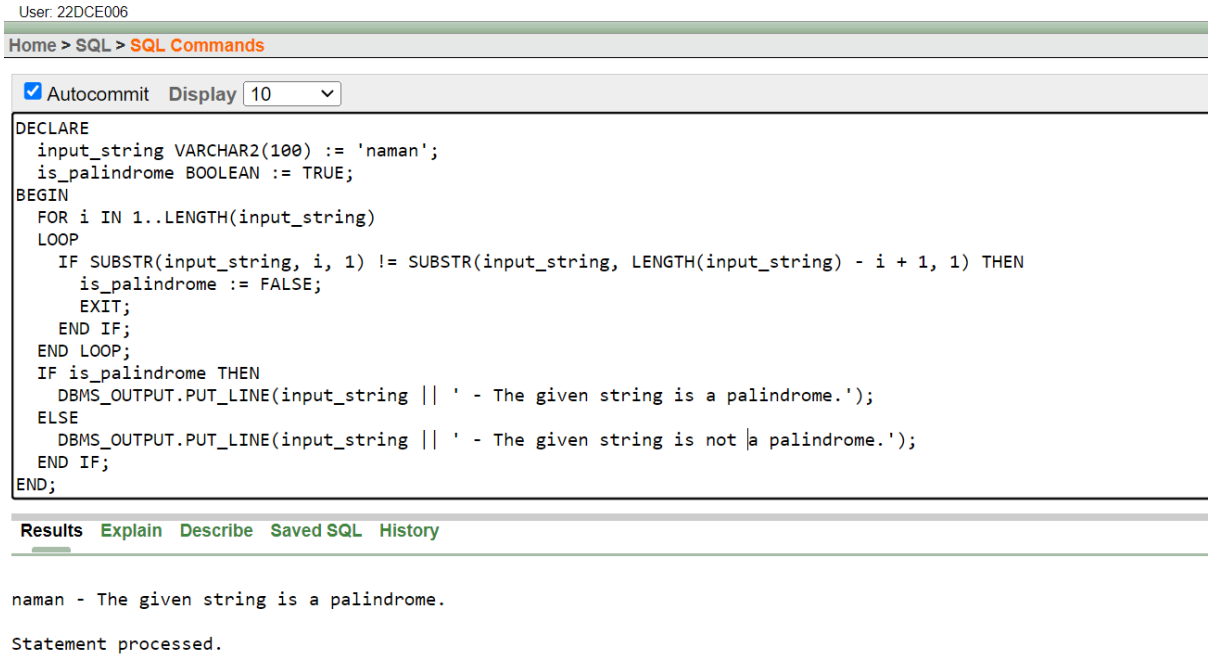
Using For Loop

Query:

```
DECLARE
  input_string VARCHAR2(100) := 'naman';
  is_palindrome BOOLEAN := TRUE;
BEGIN
  FOR i IN 1..LENGTH(input_string)
```

```
  LOOP
    IF SUBSTR(input_string, i, 1) != SUBSTR(input_string, LENGTH(input_string) - i
+ 1, 1) THEN
      is_palindrome := FALSE;
      EXIT;
    END IF;
  END LOOP;
  IF is_palindrome THEN
    DBMS_OUTPUT.PUT_LINE(input_string || ' - The given string is a palindrome.');
  ELSE
    DBMS_OUTPUT.PUT_LINE(input_string || ' - The given string is not a
palindrome.');
  END IF;
END;
```

Output:



Using While Loop

Query:

```
DECLARE
  input_string VARCHAR2(100) := 'naman';
  is_palindrome BOOLEAN := TRUE;
  i NUMBER := 1;
BEGIN
  WHILE i <= LENGTH(input_string)
  LOOP
    IF SUBSTR(input_string, i, 1) != SUBSTR(input_string, LENGTH(input_string) - i
+ 1, 1) THEN
      is_palindrome := FALSE;
      EXIT;
    END IF;
```

```
  i := i + 1;
 END LOOP;
 IF is_palindrome THEN
  DBMS_OUTPUT.PUT_LINE(input_string||' - The given string is a palindrome.');
 ELSE
  DBMS_OUTPUT.PUT_LINE(input_string||' - The given string is not a palindrome.');
 END IF;
END;
```

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
DECLARE
  input_string VARCHAR2(100) := 'naman';
  is_palindrome BOOLEAN := TRUE;
  i NUMBER := 1;
BEGIN
  WHILE i <= LENGTH(input_string)
  LOOP
    IF SUBSTR(input_string, i, 1) != SUBSTR(input_string, LENGTH(input_string) - i + 1, 1) THEN
      is_palindrome := FALSE;
      EXIT;
    END IF;
    i := i + 1;
  END LOOP;
  IF is_palindrome THEN
    DBMS_OUTPUT.PUT_LINE(input_string||' - The given string is a palindrome.');
  ELSE
    DBMS_OUTPUT.PUT_LINE(input_string||' - The given string is not a palindrome.');
  END IF;
END;
```

Results Explain Describe Saved SQL History

```
naman - The given string is a palindrome.

Statement processed.
```

Conclusion: From this practical I learned about basic pl/sql functions and use of loops.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 11

Aim: To understand the concept of "select into" and "% type" attribute.

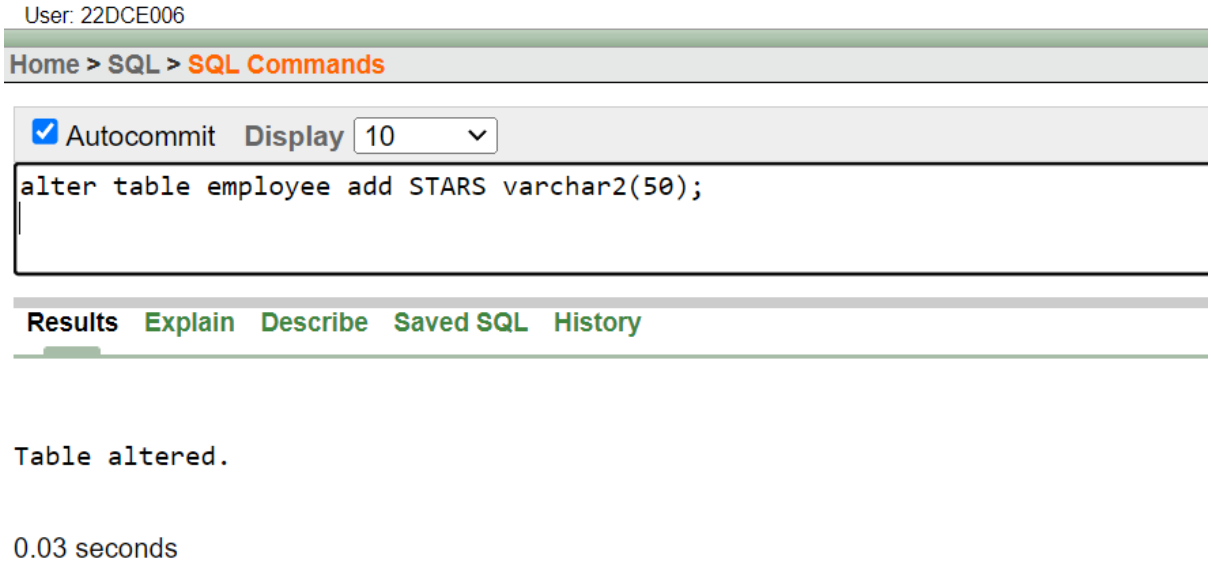
Create an EMPLOYEES table that is a replica of the EMP table. Add a new column, STARS, of VARCHAR2 data type and length of 50 to the EMPLOYEES table for storing asterisk (*).

Create a PL/SQL block that rewards an employee by appending an asterisk in the STARS column for every Rs1000/- of the employee's salary. For example, if the employee has a salary amount of Rs8000/-, the string of asterisks should contain eight asterisks. If the employee has a salary amount of Rs12500/-, the string of asterisks should contain 13 asterisks. Update the STARS column for the employee with the string of asterisks.

i)

Query: alter table employee add STARS varchar2(50);

Output:



ii)

Query:

```
declare
star_sal number;
itr number;
no number := 101;
nofstars varchar2(20);
begin
while no < 108 loop
select emp_sal into star_sal from employee where emp_no = no;
itr := ceil(star_sal/1000);
for i in 1..itr loop
nofstars := nofstars ||
'*';http://127.0.0.1:8080/apex/f?p=4500:1003:3486766455597079::NO:1003::#
end loop;
update employee set stars=nofstars where emp_no = no;
```

```
nofstars := '';
no := no +1;
end loop;
end;
select *from employee
```

Output:

Home Logout Help

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```
end;
select *from employee
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE	PHONE	INCREASE	DIFFERENCE	STARS
101	smith	800	-	10	shah	machine learning	fi_mgr	toronto	105	09-AUG-96	-	920	120	*
102	snehal	1600	300	25	gupta	data science	lec	las vegas	-	14-MAR-96	-	1840	240	**
103	David	1100	500	20	wales	machine learning	mk_mgr	ontario	105	30-NOV-95	-	1265	165	**
104	aman	3000	-	10	sharma	virtual reality	comp_op	mexico	12	02-OCT-97	-	3450	450	***
105	anita	5000	50000	10	patel	big data analytics	comp_op	germany	107	01-JAN-98	-	5750	750	*****
106	sneha	2450	24500	10	joseph	big data analytics	fi_acc	melbourne	105	26-SEP-97	-	2818	368	***
107	anamika	2975	-	30	jha	artificial intelligence	it_prog	new york	-	15-JUL-97	-	3421	446	***

7 rows returned in 0.00 seconds CSV Export

Conclusion: From this practical I learned about the concept of "select into "query ,"%type" attribute and its implementation.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 12

Aim: To perform the concept of cursor

(a) Display all the information of EMP table using %ROWTYPE.

(b) Create a PL/SQL block that does the following:

In a PL/SQL block, retrieve the name, salary, and MANAGER ID of the employees working in the particular department. Take Department Id from user.

If the salary of the employee is less than 1000 and if the manager ID is either 7902 or 7839, display the message <<last name>> Due for a raise. Otherwise, display the message <<last_name>> Not due for a raise.

(c) In a loop, use a cursor to retrieve the department number and the department name from the DEPT table for those departments whose DEPT_ID is less than 100. Pass the department number to another cursor to retrieve from the EMP table the details of employee name, job, hire date, and salary of those employees whose EMP_NO is less than 7566 and who work in that department

i)

Query:

```
declare
 cursor c1 is select * from employee;
 v1 c1%rowtype;
begin
   open c1;
 loop
   fetch c1 into v1;
   exit when c1%notfound;
 dbms_output.put_line(v1.emp_no || ' ' || v1.emp_name || ' ' ||v1.emp_sal || ' '
||v1.emp_comm || ' ' ||v1.dept_no ||' ' || v1.l_name || ' ' ||v1.dept_name || ' '
||v1.job_id || ' ' ||v1.location || ' ' ||v1.manager_id || ' ' ||v1.hiredate);
 end loop;
 close c1;
end;
```

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
declare
 cursor c1 is select * from employee;
```

Results Explain Describe Saved SQL History

```
101 smith 800  10 shah machine learning fi_mgr toronto 105 09-AUG-96
102 snehal 1600 300 25 gupta data science lec las vegas  14-MAR-96
103 David 1100 500 20 wales machine learning mk_mgr ontario 105 30-NOV-95
104 aman 3000  10 sharma virtual reality comp_op mexico 12 02-OCT-97
105 anita 5000 50000 10 patel big data analytics comp_op germany 107 01-JAN-98
106 sneha 2450 24500 10 joseph big data analytics fi_acc melbourne 105 26-SEP-97
107 anamika 2975  30 jha artificial intelligence it_prog new york  15-JUL-97

Statement processed.
```

0.03 seconds

ii)

Query:

```
declare
D_no number:=:ENter_no;
 cursor c2 is select emp_name,emp_sal,manager_id from employee where
DEPT_NO=D_no;
 v2 c2%rowtype;
begin
   open c2;
 loop
   fetch c2 into v2;
 if v2.emp_sal<1000 and (v2.manager_id = 7902 or v2.manager_id = 7839)
then
       dbms_output.put_line(v2.emp_name|| ' It is for raise.');
 else
  dbms_output.put_line(v2.emp_name|| ' It is not for raise.');
 end if;
 exit when c2%notfound;
 end loop;
```

```
 close c2;
end;
```

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
    fetch c2 into v2;
 if v2.emp_sal<1000 and (v2.manager_id = 7902 or v2.manager_id = 7839) then
        dbms_output.put_line(v2.emp_name|| ' It is for raise.');
 else
  dbms_output.put_line(v2.emp_name|| ' It is not for raise.');
 end if;
 exit when c2%notfound;
 end loop;
 close c2;
end;
```

Results Explain Describe Saved SQL History

```
It is not for raise.
```

iii)

Query-Output:

```
declare
cursor c1 is select dept_no, dept_name from employee where dept_no < 100;
cursor c2(d_id employee.dept_no%type) is select emp_no, emp_name, emp_sal, job_id, hiredate from employee where emp_no < 7566 and dept_no = d_id;

dept_id varchar(10);
dept_name employee.dept_name%type;
emp_no employee.emp_no%type;
emp_name employee.emp_name%type;
emp_sal employee.emp_sal%type;
job employee.job_id%type;
hire_date employee.hiredate%type;

begin
```

Results Explain Describe Saved SQL History

```
Department: 10-machine learing
Department: 25-data science
Department: 20-machine learing
Department: 10-virtual reality
Department: 10-big data analysis
Department: 10-big data analysis
Department: 30-artificial intelligence

Statement processed.
```

0.00 seconds

Conclusion: By performing this practical, I got to learn about concept of %rowtype and how to create cursor and use it to fetch the original table information using the cursor in PL-SQL Block.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 13

Aim: To solve queries using the concept of View.
(1) Write a query to create a view for those employees belongs to the location New York.
(2) Write a query to create a view for all employee with columns emp_id, emp_name, and job_id.
(3) Write a query to find the salesmen of the location New York who having salary more than 3000.
(4) Write a query to create a view to getting a count of how many employees we have at each department.

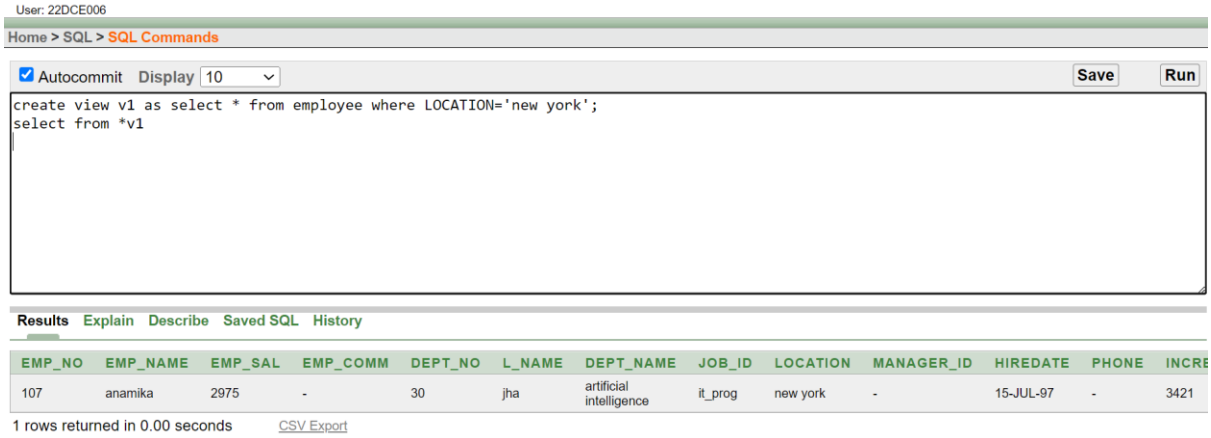
i)

Query:

create view v1 as select * from employee where LOCATION='new york';

select from *v1

Output:



ii)

Query:

create view v2 as select emp_no,emp_name,job_id from employee

select from *v2

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
create view v2 as select emp_no,emp_name,job_id from employee
select from *v2
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	JOB_ID
101	smith	fi_mgr
102	snehal	lec
103	David	mk_mgr
104	aman	comp_op
105	anita	comp_op
106	sneha	fi_acc
107	anamika	it_prog

7 rows returned in 0.00 seconds CSV Export

iii)

Query:

create view v3 as select emp_name from employee where location='new york' and emp_sal>3000;

select from *v3;

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
create view v3 as select emp_name from employee where location='new york' and emp_sal>3000;
select from *v3;
```

Results Explain Describe Saved SQL History

no data found

iv)

Query:

create view v4 as select dept_name,count(dept_name) as count from employee group by dept_name;

select from *v4;

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
create view v4 as select dept_name,count(dept_name) as count from employee group by dept_name;
select from *v4;
```

Results Explain Describe Saved SQL History

DEPT_NAME	COUNT(DEPT_NAME)
big data analytics	2
artificial intelligence	1
machine learning	2
virtual reality	1
data science	1

5 rows returned in 0.00 seconds CSV Export

Conclusion: By performing this practical, I got to learn about the concepts of view in SQL. Also, got to understand about its use and how to create it.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 14

Aim: To perform the concept of function and procedure

To update the salary of employee specified by empid. If record exist then update the salary otherwise display appropriate message. Write a function as well as procedure for updating salary.

Query:

```
create or replace function update_sal(new_sal in number)
return number
is new_s number;
begin
new_s:= new_sal;
new_s:=new_s+(new_s*0.1);
return new_s;
end;

declare
id number:= :Enter_emp_id;
a number;
b number;
begin
select emp_sal into a from employee where emp_no=id;
b:=update_sal(a);
if b=a then
dbms_output.put_line('Salary not updated');
else
dbms_output.put_line('Salary updated'||b||'with emp_id'||id);
end if;
end;
```

Output:

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
begin
new_s:= new_sal;
new_s:=new_s+(new_s*0.1);
return new_s;
end;
```

Results Explain Describe Saved SQL History

```
Function created.
```

User: 22DCE006

Home > SQL > SQL Commands

Autocommit Display 10

```
begin
select emp_sal into a from employee where emp_no=id;
b:=update_sal(a);
if b=a then
dbms_output.put_line('Salary not updated');
else
dbms_output.put_line('Salary updated'||b||'with emp_id'||id);
end if;
end;
```

Results Explain Describe Saved SQL History

```
Salary updated1210with emp_id103

Statement processed.
```

Conclusion: From performing this practical, I got to learn about use of function and procedure.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 15

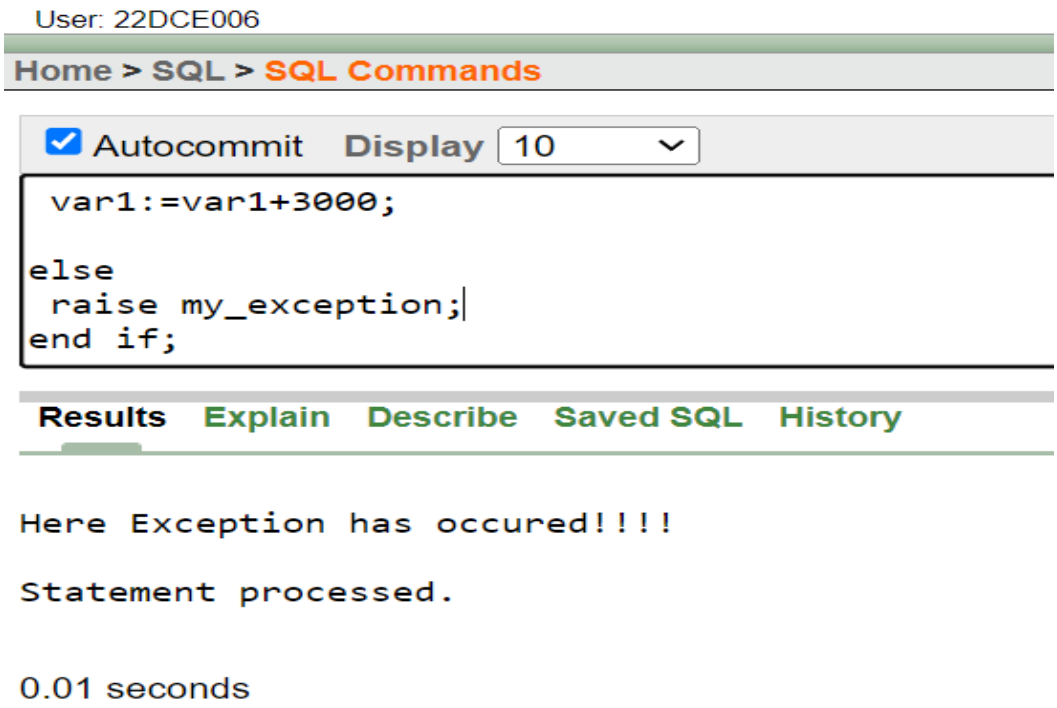
Aim: To perform the concept of exception handler

Write a PL/SQL block that will accept the employee code, amount and operation. Based on specified operation amount is added or deducted from salary of said employee. Use user defined exception handler for handling the exception.

Query:

```
declare
var1 number;
my_exception exception;
begin
select emp_sal into var1 from employee where emp_no=103;
if var1>2000 then
 var1:=var1+3000;
else
 raise my_exception;
end if;
exception
 when my_exception then
  dbms_output.put_line('Here Exception has occured!!!!');
end;
```

Output:



Conclusion: From this practical I learned the concept of exception handling using PL/SQL block.

Staff Signature:

Grade:

Remarks by the Staff:

Practical 16

Aim: To perform the concept of trigger

Write a PL/SQL block to update the salary where deptno is 10. Generate trigger that will store the original record in another table before updation take place.

Query:

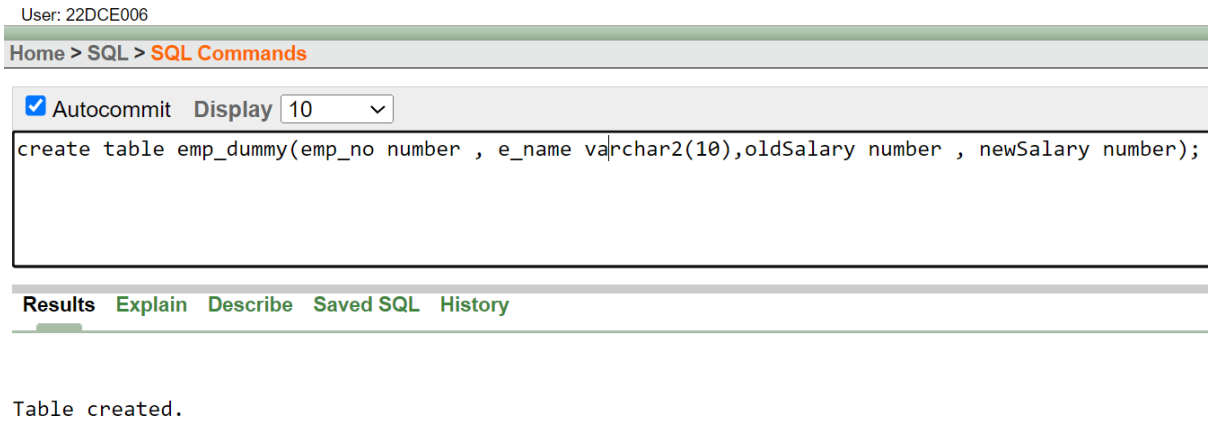
create table emp_dummy(emp_no number , e_name varchar2(10),oldSalary number , newSalary number);

```
create or replace trigger trigger_used
before update on employee
for each row
when (new.dept_no = 10)
begin
   insert into emp_dummy (emp_no, e_name, oldSalary, newSalary)
   values (:old.emp_no, :old.emp_name, :old.emp_sal, :new.emp_sal);
end;

update employee SET emp_sal = 10000 WHERE emp_no = 105 AND dept_no = 10;
update employee SET emp_sal = 5000 WHERE emp_no = 105 AND dept_no = 10;
update employee SET emp_sal = 5000 WHERE emp_no = 106 AND dept_no = 10;

Select * from emp_dummy;
```

Output:



```
create table emp_dummy( emp_no number, e_name varchar2(10), oldSalary number, newSalary number);

create or replace trigger trigger_used
before update on employee
for each row when(new.dept_no = 10)
begin
insert into emp_dummy(emp_no, e_name, oldSalary, newSalary) values (:old.emp_no, :old.emp_name, :old.emp_sal, :new.emp_sal);
end;

update employee set emp_sal = 10000 where emp_no = 105 and dept_no = 10;
update employee set emp_sal = 5000 where emp_no = 105 and dept_no = 10;
update employee set emp_sal = 5000 where emp_no = 106 and dept_no = 10;

select * from emp_dummy;
```

Results Explain Describe Saved SQL History

EMP_NO	E_NAME	OLDSALARY	NEWSALARY
105	Anita	5000	10000
105	Anita	10000	5000
106	Sneha	2450	5000

3 rows returned in 0.00 seconds CSV Export

Conclusion: From this practical I learned the concept of triggers in PL/SQL

Staff Signature:

Grade:

Remarks by the Staff:

Practical 17

Aim: To perform the concept of package

1. Create a package specification and package body called EMP_PACK that contains your NEW_EMP procedure as a public construct, and your VALID_DEPTID function as a private construct. (You can save the specification and body into separate files.)

2. Invoke the NEW_EMP procedure, using 40 as a department number. Because the dept_no 40 does not exist in the DEPT table, you should get an error message as specified in the exception handler of your procedure.

3. Invoke the NEW_EMP procedure, using an existing department ID 80.

i)Query-Output:

Home > SQL > SQL Commands

Autocommit Display 10

```
create or replace package emp_pack as
function valid_deptid(dept_id in number) return number;
procedure new_emp(p_dept_id in number);
end emp_pack;

create or replace package body emp_pack as
dept_id number;
function valid_deptid (dept_id in number) return number is
c number;
begin
select count(emp_no) into c from employee where dept_no = dept_id;
if c > 0 then
return 0;
```

Results Explain Describe Saved SQL History

```
Package created.
```

Home > SQL > SQL Commands

Autocommit Display 10

```
create or replace package body emp_pack as
dept_id number;
function valid_deptid (dept_id in number) return number is
c number;
begin
select count(emp_no) into c from employee where dept_no = dept_id;
if c > 0 then
return 0;
else
return 1;
end if;
end valid_deptid;

procedure new_emp(p_dept_id in number) is e1 exception;
begin
if valid_deptid(dept_id) = 0 then
dbms_output.put_line('Valid Department ID');
```

Results Explain Describe Saved SQL History

```
Package Body created.
```

ii)Query-Output:

Enter Bind Variables - Google Chrome

127.0.0.1:8080/apex/f?p=4500:138:4336927876133498:::

Submit

:ENTER_DEPT_ID 40

Home > SQL > SQL Commands

Autocommit Display 10

```
end if;

exception
when e1 then
dbms_output.put_line('Error! Invalid Department ID');
end new_emp;
end emp_pack;

declare
v_emp_id number;
v_emp_name varchar2(50);
v_dept_id number := :Enter_dept_id;
n number;
begin
emp_pack.new_emp(v_dept_id);

end;
```

Results Explain Describe Saved SQL History

```
Error! Invalid Department ID

Statement processed.
```

iii)Query-Output:

Enter Bind Variables - Google Chrome

127.0.0.1:8080/apex/f?p=4500:138:3312362983213307:::

Submit

:ENTER_DEPT_ID 20

Home > SQL > SQL Commands

Autocommit Display 10

```
dbms_output.put_line('Valid Department ID');
else
dbms_output.put_line('Invalid Department ID');
end if;

end new_emp;
end emp_pack;

declare
v_emp_id number;
v_emp_name varchar2(50);
v_dept_id number := :Enter_dept_id;
n number;
begin
emp_pack.new_emp(v_dept_id);

end;
```

Results Explain Describe Saved SQL History

```
Invalid Department ID

Statement processed.
```

Conclusion: From this practical I learned the concept of Packages in PL/SQL.

Staff Signature:

Grade:

Remarks by the Staff: